AD-A224 924

1987

Annual Technical Report

July 1983 - November 1987

A Research Program in Computer Technology

ISI/SR-88-255

USC

INFORMATION SCIENCES INSTITUTE

[illegible]

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT This document is approved for public release; distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) ISI/SR-88-255	5. MONITORING ORGANIZATION REPORT NUMBER(S) ———
6a. NAME OF PERFORMING ORGANIZATION USC/Information Sciences Institute 6b. OFFICE SYMBOL *(If applicable)*	7a. NAME OF MONITORING ORGANIZATION ———
6c. ADDRESS *(City, State, and ZIP Code)* 4676 Admiralty Way Marina del Rey, CA 90292-6695	7b. ADDRESS *(City, State, and ZIP Code)* ———
8a. NAME OF FUNDING/SPONSORING ORGANIZATION DARPA 8b. OFFICE SYMBOL *(If applicable)*	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER MDA903 81 C 0335
8c. ADDRESS *(City, State, and ZIP Code)* Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, VA 22209	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. ——— | PROJECT NO. ——— | TASK NO. ——— | WORK UNIT ACCESSION NO. ———

11. TITLE *(Include Security Classification)*
1987 Annual Technical Report:
A Research Program in Computer Technology (Unclassified)

12. PERSONAL AUTHOR(S) ISI Research Staff

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT *(Year, Month, Day)*	15. PAGE COUNT
Annual Technical Report	FROM 7/1/86 TO 11/30/87	1990, July	

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
FIELD	GROUP	SUB-GROUP	
09	02		(over)

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

This report summarizes the research performed by USC/Information Sciences Institute from July 1, 1986, to November 30, 1987, for the Defense Advanced Research Projects Agency. The research is focused on the development of computer science and technology, which is expected to have a high DoD/military impact. Keywords: Computer Systems; Military Research; Computer systems; Technology forecasting; Computer programs. (CP) ←

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Victor Brown Sheila Coyazo	22b. TELEPHONE *(Include Area Code)* 213/822-1511 | 22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted.
All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

18. SUBJECT TERMS (continued)

1. *CLF:* artificial intelligence, automated implementation, Common LISP, expert systems, formal specification
2. *EES:* domain model, domain principles, expert systems, integration knowledge, LISP, NIKL paraphraser, optimization knowledge, program writer
3. *FSD:* artificial intelligence, automatic programming, consistency maintenance, formal specification, programming environments, rapid prototyping, rule-based programming, software evolution
4. *ADV-VLSI:* CMOS/Bulk fabrication, MOSIS, printed circuit board fabrication, scalable design rules, VLSI
5. *VLSI:* design rules, device fabrication, device testing, integrated circuits, MOSIS, silicon compilation, VLSI design, wafer testing
6. *KITSERV:* integrated circuit design, VLSI test equipment
7. *INC:* computer communication, electronic mail, internetwork protocols, protocol design
8. *MMC:* computer communication, multimedia conferencing, packet video, packet voice, protocols, video conferencing
9. *Pro Acc:* computer communication, high-speed communication, protocols, protocol design
10. *SWC:* computer communication, protocols, protocol design, supercomputers, Wideband Satellite Network
11. *KREP:* classification-based reasoning, KL-ONE, KL-TWO, knowledge representation, Loom, NIKL
12. *Nat Lang:* Janus, knowledge representation, natural language generation, Penman, text generation
13. *Comm Mail:* computer mail, electronic mail, Intermail, mail forwarding, mail system interconnection
14. *Comp:* computer hardware support, computer operations, computer support, network services, software support
15. *DARPA HQ Support:* computer communication, hardware maintenance, remote maintenance, remote monitoring
16. *EWS:* computer workstations, distributed processing, survivable networks
17. *NCE:* distributed processing, local networks, personal computers, workstation environment
18. *SC Dev:* computer acquisition, Strategic Computing Program
19. *C3 Exp Supp:* C3, computer communication, packet radio, survivable networks

ISI/SR-88-255

University of Southern California

1987 ANNUAL TECHNICAL REPORT

July 1986 – November 1987

A Research Program in Computer Technology

Principal Investigator and Executive Director:
Keith W. Uncapher

Prepared for the Defense Advanced Research Projects Agency
Effective date of contract 1 July 1981
Contract expiration date 30 November 1989
Contract No. MDA 903 81 C 0335
ARPA Order 4242

INFORMATION SCIENCES INSTITUTE ISI

213/822-1511

4676 Admiralty Way/Marina del Rey/California 90292-6695

This research is supported by the Defense Advanced Research Projects Agency under Contract No. MDA903 81 C 0335. Views and conclusions contained in this report are the authors' and should not be interpreted as representing the official opinion or policy of DARPA, the U.S. Government, or any person or agency connected with them.

CONTENTS

SUMMARY

This report summarizes the research performed by USC/Information Sciences Institute from July 1, 1986, to November 30, 1987, for the Defense Advanced Research Projects Agency. The research is focused on the development of computer science and technology, which is expected to have a high DoD/military impact.

The ISI program consists of 19 research areas:

Common LISP Framework: producing an exportable version of the FSD testbed, which incorporates the well-understood portions of our research in FSD into a new automated software development paradigm, and distributing it to outside users; *Explainable Expert Systems*: creating a framework for building expert systems that enhances an expert system's explanatory capablities (allowing it to explain and justify its reasoning), and eases modification and maintenance of an expert system; *Formalized System Development*: studying a new automated software development paradigm in which a formal operation specification, used as a rapid prototype, is evolved to have the intended functionality, and is mechanically transformed (with human guidance) into an efficient implementation; *Advanced VLSI*: providing access to 1.2 micron CMOS/Bulk fabrication and to scalable design rules, to allow use of this emerging technology as it is developed; *VLSI*: providing a low-cost, fast-turnaround LSI/VLSI device fabrication service to support a geographically distributed VLSI research community with no direct access to VLSI fabrication but with access to a computer communication network, and conducting research on the VLSI design problem; *KITSERV VLSI Design Service*: developing cost-effective test devices for the DARPA VLSI design community, and transferring the technology to commercial companies willing to market and support the testers; *Internet Concepts*: exploring aspects of protocols for the interconnection of computer communication networks, specifically the design and implementation of new internetwork applications and services (such as the Domain Name System) and the design and analysis of internetwork host and gateway protocols; *Multimedia Conferencing*: designing and developing an experimental multimedia real-time conferencing system based on packet-switched network technology, with the goal of enabling users to communicate interactively via workstations and the Internet in a combination of text, bitmap images, and voice media; *Protocol Accelerator*: designing and implementing architectures that integrate computer and communications systems, focusing on techniques that allow standard protocols to perform at speeds comparable to the fastest communication links and applications; *Supercomputer Workstation Communication*: providing protocols and prototype programs for effective use of high-capacity communication systems such as the Wideband Satellite Network for applications involving remote access to supercomputers from powerful workstations; *Empirically Valid Knowledge Representation*: developing a language for programming intelligent applications, which will emphasize the construction of declarative models of application domains, with the aim of creating sharable and reusable knowledge bases; *Natural Language Generation*: developing new methods for autonomous creation of text by machine, with the focus on fluent, easily controlled sentence and paragraph production, faithful representation of information from AI knowledge bases, and use of generated English in ongoing human-computer interaction; *Commercial Mail*: developing and operating a service to support the exchange of electronic mail between the Internet and various commercial mail suppliers; *Computer Research Support*:

operating reliable computing facilities and continuing the development of advanced support equipment; *DARPA Headquarters Support Center*: establishing a fully operational computer facility and a large conference room at the DARPA-ISTO offices in Washington, D.C., and providing remote monitoring and maintenance of the equipment; *Exportable Workstation Systems*: developing a remote testbed environment of advanced workstations and servers; *New Computing Environment*: exploring, determining, and implementing the next generation of computers and computing facilities for the ISI research environment; *Strategic Computing Development Systems*: providing development computers for the DARPA Strategic Computing program, system integration as required, and distribution mechanisms for disseminating the systems to the program participants; *Strategic C3 System Experiment Support*: participating in a Strategic Command, Control, and Communication systems experiment demonstrating and evaluating the use of new technologies (such as the Internet, packet radio, network security, and distributed knowledge-based techniques).

1. COMMON LISP FRAMEWORK

Research Staff:	*Support Staff*:
Robert M. Balzer	Audree Beal
David S. Wile	Carol Sato
Dennis Allard	Jeanine Yamazaki
Richard Berman	
Ben Broder	
Chloe Holg	
Brent Miller	
Donald Voreck	
William Vrotney	

1.1 ACCOMPLISHMENTS

The major accomplishments of this project during the reporting period include reengineering of several ad hoc components of the CLF and incorporation of new functionality. In the former category, the CLF was ported to TI Explorers and major portions to HP Bobcats. This considerably expands the availability of the CLF within our group. The X-Windows system was integrated with the HP Bobcat system, providing a portable presentation facilty for CLF data to the user.

A major reengineering accomplishment is the incorporation into the CLF of our "world"-based mechanism for maintaining persistence of objects across crashes. This mechanism will ultimately supplant the ad hoc version for maintaining persistencef that is presently in place in the CLF. It is based on the notion that certain *aggregates* of information characterize concerns in the virtual database, concerns such as mail handling, appointment scheduling, program development, employee data, etc. The myriad types and relations involved in such concerns are grouped together in what we call a "world specification." Based on this specification the system generates a *closure algorithm* for world instances said to satisfy the specification. Initial seed objects are given to the closure algorithm, which then determines the contents of the world instance in the database. This collection of data is then made persistent, i.e., stored on disk for retrieval at a subsequent time. This mechanism is incremental, allowing it to be used for crash protection and garbage elimination. The facility is unique in that concerns do not partition the database in a "cookie cutter" fashion, as do other aggregation methods. Instead, individual objects may participate in several concerns at once. Only those aspects relevant to the concern are grouped in the world instance.

Under previous DARPA and NSF support we developed a system called Popart, a grammar-based suite of tools for program manipulation and support. This year we ported POPART to the CLF for use by arbitrary Common LISP programs. Given a

BNF-like grammar (extended with regular expression and precedence facilities), this facility provides a lexical analyzer, parser, pattern matcher, pretty printer, structure editor, semantics definition package, and even a transformation system for the language specified by the grammar. This facility will form the foundation for the FSD transformational development service in the future.

Finally, in a community service capacity, we gathered voluminous test program suites for the validation of Common LISP programs; a database for their retrieval was designed and populated and a test engine built for their use. These will be valuable for validating vendors' compilers in the future.

2. EXPLAINABLE EXPERT SYSTEMS

Research Staff:
William R. Swartout
Robert Neches
Steve Smoliar

Research Assistants:
Johanna Moore
Jody Paul

Support Staff:
Audree Beal

2.1 TECHNICAL SUMMARY

Expert systems will play an important and expanding role in the military in the near future -- they are now beginning to be included in RFPs. However, even the best expert system is worthless if it is not accepted by its intended users. A critical factor for user acceptance of expert systems is that they must be able to explain their reasoning. For an expert system to continue to be useful, it must also be possible to evolve and extend it to reflect changing needs. The Explainable Expert Systems (EES) project has been concerned with building a framework for constructing expert systems that both provides them with better explanatory capabilities and eases their maintenance. The explanations that current expert systems can provide are limited because those systems only represent enough knowledge to *solve* a problem. The knowledge needed to *justify* the system is the knowledge that was needed to design it. Because that design knowledge is not represented, good justifications cannot be provided. The EES framework is based on the observation that providing an adequate range of explanations requires not only knowledge of the expert system itself, but also knowledge of how that expert system was designed.

The EES approach to capturing the design knowledge that underlies an expert system is unique. An automatic program synthesizer is used to create an expert system from an abstract specification that includes domain-dependent and independent knowledge. As it creates the expert system, the program writer records the design decisions it makes in a machine-readable *development history*. Subsequently, this record can be examined by explanation routines to explain not only the system's actions but also the *rationale* behind those actions. Subsequent modification of the expert system is easier because it is performed at the *specification* level and the program writer is then used to re-derive a new implementation.

2.1.1 Principal Expected Innovations

- Development of a framework for building expert systems that captures the design rationale underlying the system, and makes it available for explanation.
- Development of a knowledge base architecture that eases the evolution and maintenance of an expert system by encouraging the explicit separation and

representation of different kinds of knowledge, such as domain-descriptive knowledge, problem-solving knowledge, and terminology. This separation of knowledge will increase the modularity of an expert system, allowing parts of the system to be modified independently and easing the reuse of knowledge across expert systems.

- Development of an explanation-generation facility that uses explicit *explanation strategies* to plan explanations that address users' needs. The explanation plan will be retained after the explanation is presented to aid in producing follow-up explanations in the event that the initial explanation is not understood.

2.2 SUMMARY OF ACCOMPLISHMENTS

During FY85 the basic architecture for EES was designed, the kinds of knowledge needed to support explanation were identified, and representations were created for them. A paper on EES was presented at IJCAI-85 [2] and another at NCC; the latter paper was later re-published in the journal *Future Computing Systems.* [4]

During FY86 the first version of EES was implemented. EES was used to create a partial implementation of a software tool, the Program Enhancement Advisor, and a complete implementation of an expert system for diagnosing space telemetry systems. In the latter case, the EES Program Writer produced approximately 10 pages of LISP code to implement the system. Five papers on EES and related work were published, including one in *IEEE Transactions on Software Engineering* [3] and two in AAAI-86.

During FY87, EES version II was designed and implemented; it is capable of deriving problem-solving knowledge directly from declarative domain knowledge and employs a better representation of knowledge. Version II represents and employs generic (non-domain-specific) problem-solving methods in addition to domain-specific methods.

Also during FY87, the EES Explanation Planner was designed and implementation was begun. This facility will *plan* explanations using explicit explanation strategies. This approach to explanation will make it easier to extend the explanation facility and allow it to provide follow-up explanations to clarify misunderstood explanations. Three papers on EES and related work were published; one of these will appear in a special issue of the *Journal of Expert Systems* on expertise, another will appear in a book to be published by Springer-Verlag. In addition, Swartout co-authored (with Kathy McKeown) an article on "Text Generation and Explanation" that will appear in the *Annual Review of Computer Science.* Requests for the EES code have been received from the University of Pennsylvania and the University of Colorado at Boulder.

SELECTED PUBLICATIONS

1. Mostow, J., and W. Swartout, "Towards explicit integration of knowledge in expert systems: An analysis of MYCIN's therapy selection algorithm," in *Proceedings of the National Conference on Artificial Intelligence*, 1986.

2. Neches, R., W. Swartout, and J. Moore, "Explainable (and maintainable) expert systems," in *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pp. 382-389, IJCAI-85, August 1985.

3. Neches, R., W. R. Swartout, and J. D. Moore, "Enhanced maintenance and explanation of expert systems through explicit models of their development," *IEEE Transactions on Software Engineering* SE-11, (11), November 1985, 1337-1351.

4. Swartout, W., "Knowledge needed for expert system explanation," *Future Computing Systems*, 1986.

5. Swartout, W., and R. Neches, "The shifting terminological space: An impediment to evolvability," in *Proceedings of the National Conference on Artificial Intelligence*, 1986.

6. Swartout, W., and S. W. Smoliar, "Explaining the link between causal reasoning and expert behavior," in *Proceedings of the Symposium on Computer Applications in Medical Care*, Washington, D. C., November 1987. (Also in P. L. Miller (ed.), *Topics in Medical Artificial Intelligence*, Springer-Verlag.)

7. Swartout, W. R., and S. W. Smoliar, "On making expert systems more like experts," *Expert Systems* 4, (3), August 1987. (Also in M. Richer (ed.), *AI Tools and Techniques*, Ablex.)

3. FORMALIZED SYSTEM DEVELOPMENT

Research Staff:	*Support Staff*:
Neil Goldman	Audree Beal
Tom Kaczmarek	Jeanine Yamazaki
Don Cohen	
Michael Fox	
Kirk Kandt	
Jay Myers	
Kai Yue	

3.1 SCIENTIFIC PROGRESS

Building on a technology and software base developed in several earlier DARPA-sponsored projects, a large and relatively robust testbed has been established for specification-based software development. The testbed consists of a kernel AI operating system, support software for specification management and implementation, and a generic user interface. The testbed has been used for much of its own maintenance for over 18 months, as well as for the specification and implementation of several administrative support software systems. The availability of these support applications has allowed the testbed to serve as the sole interface to *all* computing activity for several members of the research staff for the past year. A restricted version of the testbed, excluding the administrative support applications and some experimental programming support facilities, is supported as an externally available programming environment under the name CLF (Common Lisp Framework), a project funded by DARPA under a separate contract.

3.1.1 AI Operating System

The kernel of the FSD testbed is an operating system architecture that differs from conventional operating systems in three important facets:

- A *virtual objectbase* replaces the file system as the standard repository of data and means of interprocess communication. This provides a finer grained and more sharable representation of information than does text-based encoding. The objectbase is a collection of *relations*, each consisting of a collection of *tuples* of values. Both the set of relations and the collection of tuples belonging to a relation vary over time. Applications can define new relations and dynamically add and delete tuples from relations.
- All changes to the objectbase are monitored by a *consistency manager*. Applications, as well as the operating system itself, may specify first-order logic invariants on the objectbase. The consistency manager ensures that no

change is made to the objectbase if it would violate any declared invariant. A *repair* program may be attached to any declared invariant. When a repair program is provided, the consistency manager will execute it prior to rejecting an objectbase change that would violate the invariant. The repair program may suggest additions to the objectbase change that reestablish the invariant. In this way the objectbase changes from one consistent state to another. Changes to the objectbase are propagated through the repair programs until the objectbase is again consistent. An invariant, together with its repair program, is called a *consistency rule.* Consistency rules have proven useful for handling a variety of generic application tasks, including aspects of type-checking, error detection, software cache maintenance, and dynamic storage management.

- Each transition of the objectbase from one consistent state to another is monitored by an *automation manager.* Applications may specify trigger conditions on these transitions, using an augmentation to first-order logic notation. The augmented notation makes it possible to have portions of the trigger condition be relative to the "prior" state, and other portions relative to the "new" state. Whenever some binding of objects to variables in the trigger causes that trigger to be satisfied by a transition, a response program associated with that trigger is instantiated and queued. Programs in the queue are then executed, possibly making further transitions in the objectbase and adding new programs to the queue. When the queue empties, the main process is allowed to continue. A trigger condition, together with its response program, is called an *automation rule.* Automation rules have proven useful for transferring responsibility for recurring user activities to the system.

Figure 3-1 depicts this architecture. The operating system and its data reside in the virtual address space of a single workstation. Sharing of data among multiple workstations is currently accomplished by exporting portions of one workstation's objectbase and then importing that information into the objectbase of other workstations. This export/import mechanism preserves the identity of objects by locating exported objects in the importing world if they already exist there.

3.1.2 Specification-Based Software Development Support

The FSD testbed supports the construction and maintenance of Common LISP application programs. This support falls into two categories: language extensions and programming environment.

Specifications (programs) written in the testbed environment may use not only full Common LISP in their source code, but also extensions that provide access to the virtual objectbase, consistency rules, and automation rules. These programs may include compiler *annotations* of two kinds. The first sort directs the compiler to use particular data structures, from a library of many alternatives, as the means of implementing the logical relations used in the specification. The second sort provides

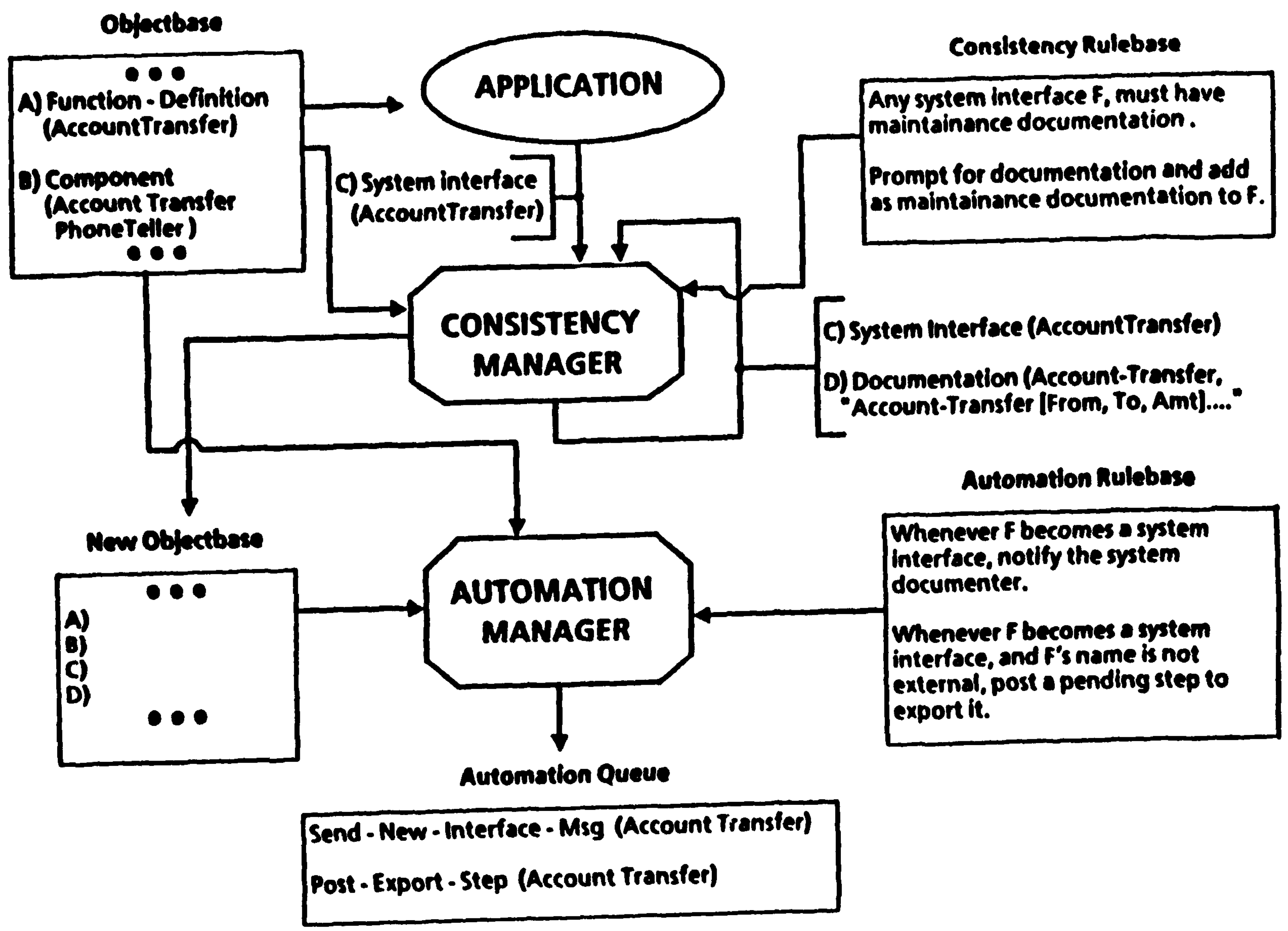

Objectbase
• • •
A) Function - Definition (AccountTransfer)
B) Component (Account Transfer PhoneTeller)
• • •
APPLICATION
C) System interface (AccountTransfer)
CONSISTENCY MANAGER
Consistency Rulebase
Any system interface F, must have maintainance documentation.
Prompt for documentation and add as maintainance documentation to F.
C) System Interface (AccountTransfer)
D) Documentation (Account-Transfer, "Account-Transfer [From, To, Amt]...."
New Objectbase
A)
B)
C)
D)
AUTOMATION MANAGER
Automation Rulebase
Whenever F becomes a system interface, notify the system documenter.
Whenever F becomes a system interface, and F's name is not external, post a pending step to export it.
Automation Queue
Send - New - Interface - Msg (Account Transfer)
Post - Export - Step (Account Transfer)

Figure 3-1: FSD logic engine

the compiler with estimates of the runtime size of these relations. Using these estimates, together with knowledge of the cost of various kinds of access for each representation in the library, permits the compiler to optimize the code it generates as an implementation of the specification. This extended language is named AP5.

The FSD testbed presents the programmer with a view of specifications as collections of *definition* objects and *system construction* code. The definitions define terms (function names, variable names, etc.) that are used as part of other definitions in the system construction code, and possibly in other specifications. The construction code provides procedural instructions for whatever is necessary, beyond loading implementations of the definitions, to establish an initial environment that is an implementation of the specification.

These objects are hierarchically combined into modules through a *component* relation. The individual objects, as well as the modules, can have a variety of assertions made about them that are used by different aspects of the programming environment. Some of these assertions are *inherited* through the component hierarchy. For example, to parse the text associated with a definition or construction code object, Common LISP's parser (the function READ) requires a "readtable," a "read base," and a "package."[1] This context is controlled on a file-by-file basis in conventional LISP programming environments. In FSD, it is controlled by assertions about the individual objects or modules that comprise the specification. Other assertions about modules control the *order* in which components of the module are displayed when it is edited, and the order in which the implementations of the components are installed when the software is loaded.

The root of such a hierarchy is called a *system*. Associated with the system is a *development record*, which records both the chronological development of the specification and plans for future development. This record consists of a hierarchical composition of steps, intended to reflect a goal-subgoal hierarchy. Each step includes an informal (textual) explanation its purpose, provided by the maintainer. At the leaves of this hierarchy are steps composed of collections of changes to the components of the system, such as additions of new definitions. The maintainer of a system extends the hierarchy with new steps when he chooses, and selects which step is to be "active." As he then makes changes to the components of his system to achieve the goal of the active step, the changes are automatically recorded. If he makes a change when no step is active, a new step is created and he is prompted for an explanation. The development record serves three purposes:

[1]The "readtable" is roughly analogous to the lexical scanner of traditional programming language parsers. The "read base" determines the radix for mapping digit sequences to integers. The "package" determines the mapping from symbolic names to LISP symbols.

1. It permits the maintainer to view, or even revert to, earlier stages of his specification or earlier versions of selected components.
2. It encourages the maintainer to lay out structured plans for future development. He can then, at a later date, indicate his readiness to "handle" one of the planned steps and have it become the active step in his development, already incorporating the explanation he provided earlier.
3. It provides the necessary data for the automatic "patch" distribution facility. The maintainer of a system can choose to provide incremental upgrades to users of that system. It is normal for users of LISP workstations to save "core images" of environments into which several extensive software systems have been loaded and perhaps even used to produce data that is also resident in the saved environment. To reload one of these systems is often not an acceptable means of upgrading the environment, since it may be very time-consuming and might destroy data by reinitializing data structures. By choosing to *distribute* a step of the development record, the maintainer automatically makes available a loadable increment to the system implementation that users may incorporate by choosing to *update* the system in their environment.

FSD's testbed provides a *static analysis* algorithm that can be applied either selectively or automatically to the definitions that comprise the specification. This algorithm produces, as tuples in the objectbase, cross-reference information about uses of definitions by other definitions. This tool can analyze both Common LISP code and the FSD extensions. It is largely template driven, and is thus relatively easy to extend to handle new sorts of definitions and new syntactic extensions that use them.

The testbed also provides an automatic source code "fonter," which adds fonts to textual definitions, thus sharpening semantic distinctions and making the text more readable.

3.1.3 Generic User Interface

Applications written using the FSD extensions to Common LISP are automatically provided significant runtime interface support driven by the specification's domain model. Application-specific menus may also be added through declarations that associate the menu label and selection response code with specific object classes. The generic interface facilities include the following:

- *Mouse-sensitive output.* Representations of objects visible to the user (whether in "viewers" (see below) or printed by a program in a LISP interaction window) may be selected with the mouse. Any visible object may be used as input to a command being entered, used as a response to a prompt, or chosen to pop up a menu of operations that can be performed on the object.
- *Dynamic object viewers.* A particular "view" of an object (subset of its attributes and their values) may be displayed in a scrollable window. The

view is kept current as the set of attributes and values in the objectbase changes.

- *Dynamic descriptions.* A collection of classifications and attribute-value pairs may be established as a *dynamic description.* The value part of each pair may be a particular object, a string pattern, or another dynamic description. Objects that satisfy the description may be displayed in an object viewer, which is kept current as the description changes. This makes it possible to "home in" on a desired object by incremental refinements to the description.
- *Pop-up viewers.* When a visible object is selected with the mouse, a pop-up window displays every generable tuple involving that object. Any object in these tuples may then be selected, producing a pop-up viewer for it. This facility makes it easy for a user to dynamically browse the objectbase in cases where the desired "view" is not available or not known to the user. In addition to chaining from one pop-up view to another, it is possible to add tuples, delete tuples, or replace attribute values from the pop-up view.
- *Name-based searches for objects.* Objects are found whose print name matches a provided string pattern. Depending on the parameters given to the search, the space over which the search is conducted may be all instances of a class, all objects satisfying a description, or all objects in the transitive closure of a relation over some given objects (e.g., all components of some system). The string match may be applied to only the names of objects in this search space, or to the names of objects that are the values of selected attributes of objects in the search space.

3.1.4 Administrative Applications

Four administrative applications have been built and are maintained within the FSD testbed.

1. *Electronic mail* permits testbed users to receive and transmit ARPANET mail. Both incoming and outgoing messages reside in the virtual objectbase. Several users of this application have made personal extensions, via automation rules, to tailor its interface to their own preferences and to automate some of their personal message handling. A VAX-11/780 acts as a server for this application, to manage low-level communication with the ARPANET and to provide secure permanent storage for messages.
2. The *Alarm Clock* application permits users of the testbed to schedule processes for execution based on the real-time clock. Automation rules cannot be used for this purpose, because time is not currently represented in the virtual objectbase in a way that permits its use as the basis for triggering activities.
3. The *ISI Personnel Database* is an open-ended application to maintain personnel and contract data related to ISI. This application is currently used to maintain in the objectbase the associations between ISI employees, the "mailboxes" where they receive electronic mail, and their office assignments, phone extenstions, and project assignments. Individual testbed users use this application to store similar information about non-ISI

employees with whom they frequently communicate. The electronic mail application uses this data, when present, to give messages the appearance of being from and to individual people rather than network mailboxes.

4. The *Scribe Interface* application provides linkage between the FSD objectbase and a Scribe document preparation system running remotely on a VAX-11/780. This application permits writers to build documents hierarchically from smaller pieces of text and edit at any level of the hierarchy. A document can be submitted to the Scribe compiler, and the compiled output printed on the desired output device, from the FSD testbed. The application also automatically retrieves and parses error messages produced by Scribe, rendering them as objects in the objectbase and providing an easy mechanism for positioning the editor at the site of an error in the document source.

The coexistence of these administrative applications in the testbed permits them to make use of one another's capabilities, as exemplified by the electronic mail application's use of the personnel database. In addition, the specification suppport environment uses the applications to enhance its utility. Examples of this are the provision of fonted, indexed hardcopies of specification sources (via the Scribe interface) and automatically generated electronic messages about bugs from application users to maintainers.

4. ADVANCED VLSI

***Research Staff*:**
- George Lewicki
- Ron Ayres
- Jeff Deifik
- Joel Goldberg
- Wes Hansford
- Lee Richardson
- Craig Rogers
- Carl Service
- Bing Sheu
- Barden Smith
- Jeff Sondeen
- Vance Tyree

***Research Assistants*:**
- David Hollenberg
- Ming Hsu
- Wen-Jay Hsu
- Shih-Lien Lu
- Mahesh Patil
- Vijay Sharma
- Je-Hurn Shieh
- Eric Shih

***Support Staff*:**
- Barbara Brockschmidt
- Mike Curry
- Sam Delatorre
- Terry Dosek
- Kathie Fry
- Terri Lewis
- Christine Tomovich

4.1 PROBLEMS BEING SOLVED

The foundation of modern military weapons systems is electronics. As levels of integration on silicon climb higher and higher, large, complex systems are going to fit on just a handful of chips. Restricting future military systems to the use of standard parts is analogous to insisting that all future movies be composed from ten prefilmed standard scenes.

Military electronics will have to enter the world of custom and semi custom integrated circuits. Military system architects must learn to design directly on silicon, taking advantage of all the opportunities afforded by that medium and at the same time learning to avoid all the pitfalls.

A large fraction of DoD's systems contractors do not own their integrated circuit fabrication facilities. These contractors are having a very hard time obtaining custom and semi-custom fabrication for military weapons systems. The reasons for this are manyfold:

- The DoD typically needs small volumes. In fact, the DoD represents only some 5 percent of the industry's total market. The economics of the industry dictate the selling of wafers in very large volumes. The only reason a fabricator will agree to produce small volumes is in the hope of getting a customer to high-volume production.
- The DoD must buy against stringent military specifications (MIL-SPEC), which vendors must certify as having been met. The majority of the industry's customers do not require such certification. The result is that DoD does not access a large number of fabricators who have high-quality processes but who do not want to bother with the business of line and parts certification.

- The interface to the industry is complex. Each vendor has its own proprietary design rules, obtainable only after the signature of nondisclosure agreements. Each vendor has its own proprietary design libraries, obtainable only on the condition that designs based on these libraries be fabricated only by the vendor owning the libraries. Vendors have proprietary design systems with similar restrictive conditions applied to their usage.

  The result is that a chip designer is forced to develop a separate interface to each fabricator. There is an extremely high probability that chips designed according to a set of proprietary design rules cannot be fabricated by a multiple vendor base. This means that the production of the chip cannot be put out for bid. It means that if the single vendor supporting those design rules decides to move on to another technology, the design can no longer be fabricated and all the time invested in the design will be lost. It means that if the vendor temporarily loses his process, fabrication of the chip and then development and production of the system will be delayed.
- The cost of doing an application-specific design is very high. It typically means associating oneself with one vendor. It means paying on the order of $30K to $50K for each iteration of a prototype design. This is the amount necessary to generate masks and to buy a minimum lot of ten wafers, each of which holds from one to several hundred copies of the design. The number of copies needed, however, is not several thousand; it is nearer to several tens. Thousands of circuits are generated because that is the minimum number that can be bought.

Before the DoD can expect to benefit from the use of application-specific VLSI in its weapon systems, it must provide its systems contractor community with a much more effective interface to the U.S. semiconductor industry than the one that presently exists. Such an interface is possible, and MOSIS has developed it by defining a standard interface through which many fabricators can be accessed. This interface includes a set of non-proprietary design rules applicable to a multiple vendor base and scalable with the decreases in feature size expected over the next several years. In addition, MOSIS has developed a multiple vendor base within the semiconductor industry capable of supporting that interface. By aggregating the integrated circuit needs of a whole community and representing them as one to that vendor base, MOSIS is benefiting both the vendors and the designers. By using the MOSIS Service, a large number of designers are able to develop their designs with little or no investment of the fabricators' engineering time. Fabricators become involved in a project only when a designer is ready to go into production.

MOSIS (through this now completed contract) has drastically reduced the cost of prototyping by holding regularly scheduled and frequent runs on which a large number of users are accommodated. Minimum lots of ten wafers are bought, but these ten wafers hold ten to thirty projects instead of only one. This means that the $30K to $50K run cost can be apportioned among those ten to thirty users.

MOSIS acquires and maintains non-proprietary design libraries fabricable by a multiple vendor base and distributes those libraries to its users and to commercial CAD vendors, who install them on their systems and make them available to users. In addition, MOSIS is developing quality-assurance procedures which, as a second neutral party, it can use to qualify wafers and provide MIL-SPEC parts. This approach allows DoD access to the whole of the U.S. semiconductor industry as opposed to just the small segment willing to undergo the rigors of conforming to MIL-SPEC certification procedures.

4.2 GOALS AND APPROACH

A few years ago, it become clear that industry would transition from the NMOS technology to CMOS. The reasons for this lay mainly with reliability of devices as their sizes shrank to the limits of available lithography. Device physics indicated that devices could be made to work with channel lengths of only 0.3 microns. A look at the expected evolution of lithography technology indicated that feature sizes as small as 0.3 microns would eventually become available. The problem that surfaced, however, was that if feature sizes became smaller and if the voltage supply levels were kept unchanged, then power density would go up. Increased power density would mean higher temperatures, which in turn would mean an increased rate at which the various failure mechanisms associated with silicon would come into play. NMOS, with its resistive loads, was recognized as a power-consuming technology that would have to be replaced with complementary MOS or CMOS technologies, which involve no resistive loads.

A second realization was that DARPA systems researchers were not going to be satisfied with accessing technology that was a few years behind the leading edge. The architectures they were interested in exploring and developing required the maximum density and performance available from technology. MOSIS' traditional approach of accessing technology that was a few years behind the leading edge would not be acceptable. Instead, MOSIS designers wanted to start exploring technologies as they were being developed. They wanted to be able to go into production as soon as development of the advanced processes was complete, instead of starting their designs at that time.

In order to prepare for the imminent switch from NMOS to CMOS and to provide the DARPA community with leading-edge technology, the MOSIS Advanced VLSI program was started.

The basic idea was to develop interfaces to groups within the semiconductor industry involved in the development of the most advanced CMOS processes. Runs would be held with selected segments of the DARPA design community submitting designs to allow development of process-control monitors, design rules, and design styles years ahead of the common availabilty of the processes. The advanced processes that were accessed in this manner involved feature sizes of 1.2 microns.

Since its inception, this program has seen the following:

- development of interfaces to five vendors
- design of parametric test structures for 1.2-micron CMOS and development of the associated test code
- definition of design rules
- development of software for assembling 1.2-micron runs
- successful completion of eight runs

The first four runs of the Advanced VLSI program were fabricated by four different vendors. This experiment showed that only one vendor, GE, had its process sufficiently under control as to warrant its use by the MOSIS community. Now the MOSIS community is accessing that vendor's process. Since then, new vendors have claimed completion of the development of their processes and MOSIS is in the process of initiating or reinitiating experimental runs with them.

The most significant result of the program was not the successful completion of the runs but the methodology that MOSIS put into operation to allow its advanced community efficient access to leading-edge CMOS. The interaction with the various process development groups led to a sufficient understanding of the processing issues to allow MOSIS to formulate a set of design rules that are not only vendor independent but that also scale with feature size.

What scalable rules mean in a practical sense is that geometry of designs based on these rules can be scaled to run on 3-micron processes as well as 2-micron and 1.2-micron processes.

The formulation of a set of scalable rules reduced one of the most serious disadvantages associated with seeking to access the most advanced processes for the DARPA VLSI design community. Advanced processing usually carries quite a cost and turnaround penalty. The cost for prototyping is higher and the queues for processing are longer. With advanced processing, costs are driven up not so much because of the increases in cost per wafer, but mostly because the type of lithography required drastically reduces the number of users that can be put on a prototyping run. Smaller feature sizes require lithography based on "stepping" one small image containing only a small number of designs across the surface of a wafer. Larger feature sizes allow the whole wafer to be exposed at once, by an image previously composed to have a very large number of designs. Large feature-size runs can easily carry many tens of designers, with the cost of the run shared among them. Small feature size runs can carry only a few designs. The cost of a small feature size prototyping run per design project can easily be ten times the cost per design project in a large feature size run.

Custom development for a chip usually requires several fabrication iterations. Such developments are difficult when each iteration is extremely expensive and takes a long time to complete. Some experienced systems designers insist that, because of this, it is not possible to develop complex systems based on leading-edge technology. They claim that only mature technologies should be used, because of their attendant lower costs, faster turnarounds, availability from a large number of sources, and lower risk.

Scalable rules allow advanced systems designers to do their initial chip development iterations using higher feature size technologies, where cost is low and turnaround is fast. Only in the final development phases do they use the smallest feature size technologies. There is even the possibility that by the time the the advanced technology has been accessed it will have become mature. This is the approach that members of MOSIS advanced design community are being advised to follow.

4.3 SCIENTIFIC PROGRESS

4.3.1 Scalable Design Rules

MOSIS has distributed to its community its scalable and vendor-independent design rules for 3-micron, 2-micron, and 1.2-micron p-well, n-well, and twin-tub CMOS/Bulk. These rules exist as files that can be accessed as automatic responses to messages sent to MOSIS.

The rules are in terms of a parameter lambda that assumes different values for different feature size processes. For example, lambda is equal to 1.5 microns for 3-micron processes, 1 micron for 2-micron processes, and 0.7 microns for 1.2-micron processes. The rules are drawn rules, with all geometrical relationships such as spacings, widths, and overlaps in terms of small integer values of lambda. This and the fact that all the rules fit onto one page make them easy to remember.

Two contact design layers exist: contact-to-active and contact-to-poly. There are rules requiring sufficient separation between the two kinds of contacts in order to allow MOSIS to independently adjust contact overlaps and conductive layer widths when converting geometry from drawn to mask.

The ability to scale and to independently adjust active, poly, metal1, and metal2 overlap of contacts and vias, and the biasing of active, poly, metal1, and metal2 geometry allow MOSIS to fit its rules very close to that of a fabricator. Very little is lost by using MOSIS' scalable rules as opposed to those defined by a fabricator to fit its specific process.

4.3.2 Generic Process Specification

MOSIS has defined all the conventions necesary to accept from its community designs in 3-micron, 2-micron, and 1.2-micron p-well and n-well CMOS/Bulk. This includes the definition of the design layer extensions and the various ways in which the scalable CMOS/Bulk geometry can be transmitted. For example, designers can submit designs with the well and implant layers not identified with regard to type. Depending on whether the CMOS run is p-well or n-well, MOSIS then makes the well layer p-type or n-type and the select p or n. Another form of submission defines both p-well and n-well layers and p-select and n-select layers; MOSIS then selects which type of layer to ignore, depending on whether the fabrication run is to be p or n.

MOSIS has distributed to its community detailed wafer acceptance specifications to be used to accept CMOS/Bulk wafers from the majority of its vendors. MOSIS has developed process-control monitors to allow it to independently determine whether delivered wafers conform to previously agreed-upon wafer-acceptance specifications. In addition, MOSIS has distributed to its community SPICE models characterizing the transistors expected on its 3-micron, 2-micron, and 1.2-micron CMOS/Bulk runs.

4.4 IMPACT

The Advanced VLSI project was initiated in order to interface the DoD systems design community to leading-edge technologies. By working to make new VLSI technologies available to this design community, the Advanced VLSI project is significantly reducing the costs and delays involved in the design of prototype VLSI systems.

5. VLSI

***Research Staff*:**	***Research Assistants*:**	***Support Staff*:**
George Lewicki	David Hollenberg	Barbara Brockschmidt
Ron Ayres	Ming Hsu	Mike Curry
Jeff Deifik	Wen-Jay Hsu	Sam Delatorre
Joel Goldberg	Shih-Lien Lu	Terry Dosek
Wes Hansford	Mahesh Patil	Kathie Fry
Lee Richardson	Vijay Sharma	Terri Lewis
Craig Rogers	Je-Hurn Shieh	Christine Tomovich
Carl Service	Eric Shih	
Bing Sheu		
Barden Smith		
Jeff Sondeen		
Vance Tyree		

5.1 PROBLEM BEING SOLVED

The VLSI design communities of DARPA and NSF require access to VLSI fabrication in order to investigate design methodologies and architectures that use state-of-the-art technologies. Recognizing this need, DARPA established the MOSIS (MOS Implementation Service) system at ISI in January 1981. Under this completed contract, MOSIS has accomplished the following:

- reduced the cost of VLSI prototyping
- shortened turnaround time for VLSI prototyping
- freed designers from fabrication idiosyncrasies
- made design less dependent on specific fabrication lines

A cost reduction of one to two orders of magnitude has been achieved by sharing a single fabrication run among many users. Also, by centralizing (and computerizing) idiosyncratic knowledge about vendors, MOSIS minimizes the the time designers spend away from the design problem. Serving as the only interface between its design community and the vendor base, MOSIS is able to provide turnaround times of eight to ten weeks for standard technology runs, except when unusual fabrication problems occur. Nonstandard technologies and experimental runs generally require longer fabrication schedules.

5.2 GOALS AND APPROACH

MOSIS combines the various aspects of maskmaking, wafer fabrication, and package assembly. The major components of the MOSIS system are listed below.

- interaction with the designers
- handling of their design (CIF or GDS) files
- communication over either the ARPANET or GTE Telemail
- placement of projects on dies, and dies on wafers
- matching of MOSIS design rules to specific vendors' design rules, and addition of alignment marks, critical dimensions, and test devices
- fabrication of E-beam mask sets (via subcontract)
- fabrication of wafer lots (via subcontract)
- wafer probing and data analysis of MOSIS test structures
- generation of SPICE decks for each run
- generation of bonding maps
- wafer sawing, die packaging, and bonding (via subcontract)
- device distribution

Designers use any available design tools to create artwork (layout) files, which are sent to MOSIS via the ARPANET or other computer networks. MOSIS compiles a multiproject wafer and contracts with the semiconductor industry for mask making, wafer fabrication, and packaging. MOSIS then delivers packaged IC devices to the user. The user perceives MOSIS as a "black box" that accepts artwork files electronically and responds with packaged IC devices.

Though MOSIS may be instrumental in providing cells and design tools to the user, it is the sole responsibility of the user to see that the submitted patterns yield working designs. One may compare MOSIS to a publisher of conference proceedings compiled from papers submitted in "camera-ready" form, where the publisher's responsibility is to produce the exact image on the right kind of paper using the appropriate ink and binding--but not to address the spelling, grammar, syntax, ideas, or concepts of the various papers.

MOSIS provides a clean separation of responsibility for the "printing" of chips. The semiconductor manufacturer is responsible for the processing of the parts and must satisfy MOSIS's rigorous quality-control specifications. MOSIS is responsible to the user for the quality and timeliness of the fabrication. The user is responsible for the proper design of the parts and may use any design methods he finds appropriate for his needs.

It is quite common that very advanced and sophisticated chips fabricated by MOSIS work on "first-silicon." An example of this is Caltech's MOSAIC--this is an amazing accomplishment with the existing design tools. Unfortunately, this is done at a considerable cost; for example, it is estimated that Caltech's MOSAIC chip consumed over 1,000 CPU hours on various VAXes before it was submitted to MOSIS for fabrication.

5.3 SCIENTIFIC PROGRESS

5.3.1 Technology Base for Fabrication Runs

CMOS/Bulk

MOSIS routinely supports 3-micron (feature size) CMOS/Bulk. There are two technologies available through MOSIS at 3 microns. The first is a single-metal, double-poly process, where the second poly layer serves as an electrode for high-value capacitors. The second is a double-level metal process, with runs every other week. This process has two sets of design rules. One is specific to 3-micron p-well technology; the other is a scalable set applicable to 3-micron, 2-micron, and 1.2-micron p-well and n-well CMOS/Bulk technologies. As part of the Advanced VLSI program, MOSIS now offers fabrication at 2 and 1.2 microns. MOSIS is actively engaged in expanding its 2-micron and 1.2-micron vendor base.

NMOS

MOSIS supported NMOS at 3-micron feature sizes, with *buried*, rather than *butting*, contacts, in accordance with the Mead-Conway design rules.

Completed Fabrication Runs

The following is a categoric breakdown by technology of the fabrication runs completed during this reporting period:

12 runs	CMOS/Bulk, 2 u	245 projects (20.4 per run)	avg T/A 16.6 weeks
22 runs	CMOS/Bulk, 3 u	1363 projects (37.9 per run)	avg T/A 8.8 weeks
10 runs	NMOS, 3 u	297 projects (29.7 per run)	avg T/A 7.3 weeks

5.3.2 Fabrication Interface

The MOSIS vendor base has expanded substantially during this reporting period. Increased user feedback and more extensive test results have allowed the MOSIS project to determine and communicate fabrication requirements to new vendors. This has resulted in higher quality wafers and the development of consistently reliable vendor sources for mask making and NMOS fabrication.

MOSIS has instituted procedures to manage the vast amount of information inherent in dealing with a multi-vendor base. Many administrative tasks have been automated, including the maintenance of templates to determine fabrication requirements specific to vendor and technology (a single vendor often provides several fabrication technologies).

5.3.3 Quality Assurance/Design Interface

Most MOSIS devices are prototypes without established functional testing procedures. Generally, the designers who receive these devices are still debugging the designs, rather than checking for fabrication defects introduced by less-than-perfect yield.

MOSIS's extensive quality-assurance program is aimed primarily at the parametric level. This guarantees that the electrical properties of the wafers are within specifications established by the best a priori simulations used in the design process. Work has continued to increase the accuracy of the SPICE parameters that are made available to MOSIS users. SPICE provides simulated mathematical models for the behavior of transistors, allowing designers to assess a small digital circuit idea, to avoid faulty design, and to improve their chances of success in fabrication. The electrical criteria are a superset of the SPICE parameters at level II. They include a ring oscillator, which gives a rough idea of the speed of the resulting circuitry. The electrical properties of the wafers are extracted first by the fabricator, who uses either his own process-control monitoring devices or the MOSIS test structures. Only wafers passing these tests are delivered to MOSIS.

It is a common practice in the IC industry to save functional probing time by probing wafers in only a very few sites. This practice makes sense, because all parts are subject to functional testing and because this parametric probing serves only to eliminate disastrously bad wafers.

Since designers hand-test most MOSIS devices, MOSIS requirements for parametric testing are higher than industry standards. MOSIS inserts its own test strip on every device, if space permits. This probing provides important statistics on the electrical properties and their distribution across the wafer. Most wafers have uniform distribution; some, however, have other statistical patterns, such as significant gradients and bimodal distributions.

These in-depth statistics are available only to the fabricators. Designers receive the general statistics (mean and variance) for each run. Interested users can request the specific values of the parameters extracted near any of their chips.

Users comparing the performance of actual chips to their simulations find it useful to rerun the simulation with the actual a posteriori parameter values (SPICE deck) extracted for that run.

A perfect set of electrical parameters does not guarantee perfect yield, so there is always a need for functional testing. MOSIS does not have the facilities for high-speed functional testing, but can perform partial functional testing. This screening typically catches the majority of fabrication defects, such as shorts. The screening is performed by applying and reading user-provided vectors to each device before the wafer is cut; those failing the test will not be packaged. By screening the larger chips--which typically have lower yield and higher packaging cost, and are required in larger quantities--MOSIS significantly reduces the packaging cost.

5.3.4 Standard Pad Frame/Packaging

MOSIS's current packaging strategy is to package enough parts to ensure a 90 percent probability of delivering a defectless part to the designer. This strategy was acceptable when most of MOSIS's community was designing small circuits and the fraction of packaged defective parts was small. However, a significant portion of the community has successfully completed the development of large designs and now wants from 300 to 3,000 working parts to begin developing prototype systems based on parts obtained through MOSIS. The yield for these large designs is expected to be 25 percent at best. If MOSIS were to follow its current strategy of packaging parts without any testing to indicate functionality, it would be packaging four times the required number of parts to achieve a requested quantity.

To avoid such waste, MOSIS has worked with Stanford University to define a functional test language (SIEVE) and has developed hardware to effect the testing specified by that language. Users now have the option of submitting text describing limited test procedures to be used at wafer probe to screen out bad parts. The purpose of this screening is to detect the types of "trivial" defects that cause the majority of bad parts and, therefore, to reduce packaging costs. Full functional testing is expected to be done by the user.

For designs with custom pad layouts, it is the responsibility of the designer to provide MOSIS with the custom probe card to probe his circuits. To eliminate the inconveniences associated with generating custom probe cards for every design, MOSIS has developed a set of standard pad frames, each specifying exactly where the pads are positioned. MOSIS stocks probe cards for each of the frames.

These standard frames are also expected to facilitate packaging. Bonding diagrams for projects currently submitted are generated manually, because several attempts to automate this process have met with only limited success. Bonding diagrams instruct the packager to connect a specific pad on the chip to a specific pin on the package. Standard pad frames have standard bonding diagrams, eliminating the the need to generate a new diagram for each project. The increased use of the standard pad frames has reduced the considerable amount of time needed to manually generate bonding diagrams. Standard frames also allow the bonding process itself to be automated. Automated, programmable bonding machines are currently available. Standard pad frames make possible a scenario in which an operator would identify a first pad and package pin; programmed information would then control the bonding on a chip.

5.3.5 Standard Cells

MOSIS has acquired from DARPA a government-designed standard cell library for 3-micron p-well CMOS/Bulk, designed in rules identical to MOSIS and thus capable of being fabricated by MOSIS CMOS fabricators.

MOSIS has made this library available to its community. Moreover, MOSIS has contacted all the major commercial CAD vendors to encourage them to support the library. A number of them have decided to do so. This allows the MOSIS community to purchase turnkey CAD systems to design standard-cell-based circuits that can be fabricated by MOSIS.

5.4 IMPACT

MOSIS's main function is to act as a single interface ("silicon broker") between a geographically distributed design community and a diverse semiconductor industry. As such an interface, MOSIS has significantly reduced the cost and time associated with prototyping custom chips.

The greatest impact of MOSIS, however, is in the community it has created. The MOSIS user community shares not only the fabrication services, but also experience, cells, tools, and software. The rapid growth of the community proves that the services provided by MOSIS are useful and important to both the academic and the R&D communities.

6. KITSERV VLSI KIT DESIGN SERVICE

***Research Staff*:**	***Support Staff*:**
Robert Parker	Janna Tuckett
Robert Hines	Shorty Garza
Jeff LaCoss	Jerry Wills
Richard Shiffman	Joseph Coyazo
Bert White	

6.1 PROBLEM BEING SOLVED

The increased complexity of military command and control weapon systems relies heavily on the ability to design, prototype, and produce relatively small-volume, highly reliable microelectronic parts. The increase in the rate of obsolescence of military systems further compounds the need for design, rapid prototyping, and manufacture on demand of microelectronic circuits and systems. The ISI-based, DARPA-sponsored MOSIS brokerage service has demonstrated the ability to provide fabrication of prototype and small-volume integrated circuits on demand. This capability has created the need to develop newer methodologies for design verification and system-level prototyping.

A VLSI kit design service called KITSERV was established in 1986 to demonstrate the ability to develop and integrate a low-cost functional design verification capabilility into the DARPA VLSI design environment, and additionally, to demonstrate the system prototyping methodology based on systems kits.

6.2 GOALS AND APPROACH

The overall goal of the KITSERV project was to develop cost-effective test capability for VLSI designers as an integral part of their design environments, and to make this capability available to a large population of designers by transferring the technology to commercial companies willing to market and support the testers. Specific goals involved addressing the shortcomings of commercial test systems.

Large parametric and functional testers are available in the commercial marketplace (costing approximately $1 million each) that are designed to support production testing. These systems are extremely costly to program, are not compatible with the DARPA community test interchange language, and are not integrated into the design environment. They are not cost-effective for prototype and small production testing. Lower cost (approximately $125,000) functional testers have recently been introduced that run at reasonable test rates, but these testers have limited test capabilities, are not oriented toward concurrent testing of many different designs, and are not easily integrated into the DARPA design environment. These low-cost testers require a hand-wired "load board" for each chip design to be tested, making them useless in the DARPA/MOSIS prototyping environment (because the MOSIS Service supports dozens of designs on each run).

There are currently no testers available, short of the largest systems described above, that can support the inherent increase in operating frequencies that will be achieved with chips fabricated by the MOSIS Service at 1.25 micron technology. Likewise, the increased design complexity that will be achieved with wafer-scale integration, coupled with these higher speeds, will require very wide, very deep, and very fast local vector storage with implied data rates in the hundreds of megabytes per second, far in excess of the capabilities of commercially available testers.

The KITSERV effort focused on developing low-cost functional design verification capabilities integrated into the DARPA design environment. A major goal of the KITSERV effort was to provide functional test support to the individual designer at a cost that is relatively small compared to his design environment. Two major development efforts were completed. SUNKIT ONE, the first of these efforts, involved the design and production of a VLSI chip tester based on an NMOS VLSI chip set developed for DARPA at Stanford University. The second, SUNKIT TWO, involved the design and production of a totally new, high-performance VLSI functional tester whose specifications were derived in part from user feedback resulting from prototype distribution of SUNKIT ONE.

6.3 SCIENTIFIC PROGRESS

The SUNKIT ONE tester connects to a host computer, such as a SUN workstation, via a DMA board installed in the host machine backplane. Test vectors written in the DARPA standard interchange language (SIEVE) are compiled on the host machine, broken down into pin directives, and sent to the tester box for execution. The VLSI chip set in the tester interprets the pin directives and addresses the appropriate device pins. The tester is housed in a custom-designed, vacuum-formed enclosure containing device sockets for all of the MOSIS Service's standard packages. Power is supplied from a remote power supply through a connecting cord to remove all AC power from the tester box. The tester features device-power-current monitoring and limiting, and provides for testing up to 128 device pins either in a packaged part or through a cable connection to a wafer probe station. A technique of "weak drive" is employed by the custom VLSI chip set in the tester to constantly drive all DUT (device under test) pins to logic high. DUT pins attempting to drive the tester input pins must overcome the weak-drive outputs of the tester. Test speeds with SUNKIT ONE are approximately 10 microseconds per device pin directive.

Driver software allows chip testing on the SUNKIT ONE tester attached to a Multibus SUN, a VME-Bus SUN, a VAX, or a Q-Bus-based PDP-11. No commercially available tester provides this level of flexibility in host support.

The SIEVE compiler written in the C language is available for UNIX machines, as well as the original VAX VMS implementation. The test environment is fully integrated into the design environment. A designer running the Berkeley design tools on a SUN workstation can design, simulate, submit designs to the MOSIS Service, and functionally test chips, all in the same integrated environment.

Several SUNKIT ONE testers have been produced and have been distributed to the DoD community as no-cost prototype loaners for evaluation. SUNKIT ONE testers have been used to debug the MIPS-X RISC processor and the SUNKIT TWO pin-drive chips. Derived user feedback was critical to the specification formulation of the design of SUNKIT TWO.

The second board-level SUNKIT TWO prototype has a 128 I/O pin capability. Local vector-fetch control and logic allow tester operation at a VLSI-limited rate of 19 megavectors per second. The hardware provides a test loop capability and an 8K vector depth. An exhaustive internal review of the SUNKIT TWO prototype included measuring performance of the hardware under actual testing conditions, identifying the limiting factors, evaluating tester software, and suggesting solutions to problems uncovered. Tests for a standard selector chip and a memory device were written, compiled, and successfully run on the tester hardware. Several problems and limitations were discovered, and those problems that could be addressed with relatively minor changes to hardware or software have been implemented. These efforts allowed for a focus on formulating a plan for commercial transfer of KITSERV products and on fabricating and evaluating a new version of the pin-drive electronics chip.

A printed circuit board was developed to provide a software development and hardware characterization platform for the MIPS-X processor chip. The board, designed to plug into a SUN workstation, contains the MIPS-X processor chip, the bus interface, and local program memory. The board logic was designed at Stanford using their THOR simulation environment. The wirelist and partslist extracted for THOR were exported to ISI as the design database. In addition to providing the SUN-format board for Stanford, this application investigated the system implementation methodology using logic design and simulation in one environment, while board design and layout were performed in a distinctly different environment. The bus interface for the MIPS-X board could be used in further design efforts as a systems kit module. Complete electronic specification of this interface exists as part of the documentation of this project.

6.4 IMPACT

Support of SUNKIT ONE testers installed at Stanford University, Syracuse University, Aerospace Corporation, the University of California, the University of Vermont, and the MOSIS project continues. These testers have been incorporated into simulation environments and have supported design verification of several DARPA projects, including MIPS-X and the SETI effort. These prototypes will continue to be circulated to NSF and DARPA sites as need and supply dictate.

The KITSERV project has been instrumental in bringing low-cost test capability to the educational community. An agreement has been reached with CADIC, a VLSI test equipment manufacturer, to supply MOSIS TinyChip educational users with CADIC Model 4100 testers at a reduced cost. The Model 4100 is functionally similar to the SUNKIT ONE and can provide relatively low-cost static test capabilities for support of education. Because a commercial equivalent is now being produced, no further development is planned on SUNKIT ONE.

As a method of assessing the commercial marketability of the SUNKIT TWO tester architecture, CADIC was chosen to evaluate SUNKIT TWO. One of the prototype systems was shipped to CADIC's Portland headquarters, and CADIC completed its evaluation of the SUNKIT TWO prototype tester. CADIC was able to perform functional tests with the prototype, correctly identifying good and bad parts. However, several concerns were raised during the evaluation process. Many of these concerns were based on marketing requirements of a more general audience than was addressed by the SUNKIT TWO development.

The performance of the SUNKIT TWO positions it in the middle of the existing CADIC product line, between the medium-performance Model 5100 and the highest performance Model 5200 -- thus competing with both for inclusion in the line. The inclusion of SUNKIT TWO in the CADIC product line would require these existing products to be redefined to create a "hole" for SUNKIT TWO. Although this possibility was discussed, it was decided that the unique features of the SUNKIT TWO implementation were not compelling enough to warrant disruption of the existing product line.

We believe that the conceptual transfer of the SUNKIT TWO implementation has been made and will dramatically affect future commercial product offerings. For this reason, we have decided not to continue further development on SUNKIT TWO.

The unique features of SUNKIT TWO include single-board implementation, CMOS VLSI pin-drive electronics, toroidal layout surrounding the DUT, per-pin architecture, and ability to plug additional boards together to increase the number of tester pins.

These features promise for inclusion in future products and are essential for higher speed operations.

6.5 FUTURE WORK

No further development will be made with SUNKIT ONE or SUNKIT TWO. However, focus for the continuing effort, under separate contract, will be in the area of specific VLSI-based pin-drive electronics development and test vector interchange format support software.

7. INTERNET CONCEPTS

Research Staff:	***Support Staff:***
Jon Postel	Celeste Anderson
Bob Braden	Jan Brooks
Danny Cohen	Kathleen Sullivan
Greg Finn	
Paul Mockapetris	
Joyce Reynolds	
Ann Westine	

7.1 PROBLEMS BEING SOLVED

The goal of the Internet Concepts project is to extend and enhance computer communications. The group's work is based on the Internet, a system of interconnected computer communication digital packet networks. The rules of communication (protocols) are the key element in successful computer communication. The Internet has working protocols for communication between heterogeneous computers via an interconnected collection of heterogeneous packet networks.

This research covers work at several levels, involving applications protocols as well as *host-to-host* and gateway-to-gateway protocols. In the applications level protocols, the focus is on new uses of the Internet based on procedures for hierarchical naming, and computer mail interoperation. The basic protocols are largely complete, but a number of extensions are being explored to integrate and extend the packet network technology.

The Internet Concepts project specializes in prototype development of new applications of computer networks. In particular, this project is concerned with the development of various prototype implementations of Internet services (e.g., the Domain Service and Computer Mail Interoperations); in addition, it provides research studies of various computer communication issues.

The growth of the Internet has led to a problem in maintaining the database of names of computers (host names), their addresses, and other related information. The Domain Name System (DNS) is a replicated, distributed database system developed to provide a basis for distributed management and maintenance of this important host name database.

The usefulness of computer mail or electronic mail in the DARPA community and in the commercial world has led to a desire for some form of interoperation between the ARPA-Mail world and some commercial mail systems. The Intermail program is an experiment in supporting such interoperation between the incompatible procedures of the commercial mail systems and the ARPA-Mail world.

Another concern raised by the growth of the Internet is the development of appropriate routing procedures for very large systems. Most of the routing procedures use table space or information exchanges that grow with the size of the system. For very large systems such procedures become unworkable. The development of the Cartesian Routing procedure is an exploration of a routing method that is independent of the size of the system.

The Internet Concepts project continues to assist in the ongoing evolution of the Internet, especially in the introduction of new capabilities. This project aids with the coordination, assignment, and maintenance of network parameters such as network numbers, protocol identifiers, and the like; and with the development and maintenance of protocol specifications and other documents relating to the operation of the Internet.

7.2 GOALS AND APPROACH

The long-term goals of the Internet Concepts project are to provide appropriate and effective designs for the primary user-service applications in the internetwork communication environment. The designs are based on a set of host- and gateway-level protocols that provide the full range of service characteristics appropriate to a wide variety of applications.

Our approach has been to pursue in parallel the abstract design and experimental implementation of protocols. The interaction of these activities provides valuable insights into problems and potential solutions. In particular, we continue to evolve and incorporate further capabilities in the Domain Naming System, experiment with protocol interoperability via the Intermail mail-relaying system, and investigate issues of addressing and routing in very large networks.

7.2.1 Domain Name System

In a large internet system, the management of names becomes a complex task. The simple approach of maintaining a central table of names, used in the past, has become unwieldy. To remove the practical and technical constraints, and to provide new functionality for new applications as well as provide for the continued growth of the Internet system, the Domain Naming System is being developed. This system provides structured names and allows subdivision of name-management duties.

Our goal is to provide a design and prototype implementation for a distributed system of domain servers that manage the access to a distributed database of names and associated data. Our approach is to create the design for the system and to review it with interested parties in the Internet research community. As a result of feedback received from these reviews, the design is modified. The operation of the prototype server provides feedback for revision of the design.

7.2.2 Computer Mail Interoperation

We are conducting an experiment in the interoperation between otherwise incompatible communication systems by developing a prototype service for transmitting computer mail between the ARPA-Mail system and three commercial mail systems: Telemail, MCI Mail, and ITT Dialcom Systems. This requires (among other things) development of techniques for addressing computer mail to destination addresses that may not be allowed (by syntax or semantic checks) in the originating mail system.

7.2.3 Protocol Studies

Many potential extensions to the Internet system have been discussed and desired but have not yet been designed, implemented, or tested. Our goal is to study some of these issues and, when possible, to provide designs for their eventual development.

The areas of addressing, routing, and multiplexing are particularly subtle and require careful attention. We have concentrated on these areas and have explored many options in technical discussions and memos. Our approach is to develop an understanding of these technical issues and to advocate the inclusion of general-purpose support mechanisms in the protocol specification. In addition, the project assists in the evolution of the Internet by managing the assignment of protocol parameters to network researchers as needed, and producing up-to-date documents of the protocols used in the Internet community on an as-needed basis.

7.3 SCIENTIFIC PROGRESS

7.3.1 Domain Name System

The prototype server continues in operation and the database has been substantially enlarged. This server is running on four machines distributed around the Internet and provides the name lookup service for the critical top levels of the name space. We released a new version of the prototype domain name server for TOPS-20 (the program is called JEEVES). It includes a resolver, which uses aged round trip times to optimize timeouts. This operational experience has provided confidence in the database delegation, caching, and timeout features that are the basis of the DNS. This is important in light of the explosive expansion of the Internet.

The use of the mail exchanger (MX) entries in the database is now fully supported. This feature allows CSNET, UUCP, and BITNET mail destinations to use normal mail addresses, and has begun to promote uniformity in mail addressing.

The DNS is also being used as a distributed database tool by other researchers.

Examples include its use by several sites for direct user (as opposed to host) registration for mail and other purposes. The services provided by the DNS have been selected for use as the foundation for the NETBIOS name service. These experiments will lead to new functionality in the DNS and will also serve as a useful stimulus for refinement of DNS mechanisms. We have provided guidance and support for many of these efforts.

The DNS client software developed for the Xerox Development Environment (XDE) in Mesa on the 8010 workstation is used as a testing facility for the domain system and to provide regular name lookups for other uses of the workstation. We made a number of performance improvements to the TCP and FTP implementations that run in the workstation. The DNS software consists of a domain name resolver that is accessed via the standard software interfaces in the workstation protocols package and by a special-purpose test tool (called NameTool). This NameTool may be used to send queries to the various domain name servers in the Internet, and to display their responses. This software was distributed to various universities that participate in the Xerox University Grant Program.

7.3.2 Computer Mail Interoperation

The evolution of large electronic mail systems testifies to the increasing importance of electronic mail as a means of communication and coordination throughout the scientific research community. These systems include the Internet mail system, GTE Telemail, MCI Mail, and IEEE Compmail (a Dialcom system). Until recently, these systems have operated autonomously, and no convenient mechanism has existed to allow users of one system to send electronic mail to users on another system. Intermail is an experimental mail forwarding system that allows users to send electronic mail across such mail system boundaries. Users on each system are able to use their usual mail programs to prepare, send, and receive messages. No modifications to any of the mail programs on any of the systems are required.

Intermail has a login account and a mailbox on each of the commercial mail systems it services. A user on a commercial system sends mail destined for a user in the ARPA-Mail world to the "Intermail" mailbox on his local commercial system. The Intermail program periodically picks up this mail, determines the destination ARPA-Mail address, and turns it over to the Internet mailer for delivery to the appropriate Internet host. In the other direction, a user on an Internet host sends mail destined for a user on a commercial mail system to the ARPA-Mail mailbox of Intermail ("INTERMAIL@ISI.EDU"). Intermail periodically picks up the messages in its mailbox, determines the destination mail systems, and sends each message using the TELEMAILER, COMPMAILER, or MCIMAILER program. The Intermail program examines the forwarding information contained in each message to determine the destination system and mailbox.

The Intermail experiment continues to provide regular service. We have implemented automatic, program-generated error messages for the Intermail mail forwarding system. Error-processing capabilities were added to the system to enable it to return error messages generated by a remote ARPANET mailer to the original sender of a message on a commercial mail system.

All of the mail systems involved have changed during the year (adding new features, etc.). This has required modifications to the Intermail programs to maintain compatibility. This year we added service for two additional Dialcom systems for NSF and ONR.

7.3.3 Protocol Studies

A routing procedure was developed that routes packets based on distance. Each packet carries the coordinates of the destination and each node routes the packet to reduce the distance remaining to the destination. This procedure is called Cartesian Routing. Properties of this procedure are: (1) it scales to very large network size, (2) it does not require large tables or routing information exchanges, and (3) it allows mobile hosts.

We continued studying the issues involved in large-scale routing. In particular, we are currently examining the resistance of network protocols to attack. If we assume software error, hardware error, or malicious attack at a single network router, what damage can occur to the network as a result? ARPANET SPF, Internet GGP, hierarchical, and Cartesian protocols are being examined to determine their attack resistance and to explore how they might be augmented to improve resistance. We continued researching recent results on the "Byzantine generals problem" to see if they can be applied toward attack-resistant routing.

We have used testing and monitoring programs based on the Xerox Dandelion workstation running the XDE or Mesa environment. We made a number of updates to the Internet protocol software used in our test environment. The INSpyTool was improved and now displays the headers on broadcast packets. Several minor display bugs and filtering bugs were fixed as well. These updates have been distributed to various universities who participate in the Xerox University Grant Program.

As part of our management of protocol parameters, we had been assigning network addresses for the individual networks in the Internet. As this activity became routine, we transferred this task to the Network Information Center (NIC) at SRI International. We continue to assign other protocol parameters and coordinate the documentation of protocol specifications and standards.

7.4 IMPACT

The Domain Naming System extends the power of the internet system to identify resources by name by increasing the flexibility and dynamics of name management. The Computer Mail Interoperability effort extends the communication capability for this one very popular mode of computer-based communication. The expanding use of the Internet system requires a continuing effort to provide up-to-date knowledge of system design, specifications, and performance.

7.4.1 Domain Name System

The Domain Name System has replaced the previous HOSTS.TXT method on most of the hosts in the Internet. The operational feasibility of the scheme has been demonstrated, and the combination of heterogeneous management and hosts, datagram access, and size make it a unique distributed database.

The DNS has been used to enhance the capabilities of the existing mail system through the mail exchanger records for mail repositories and routing. Several groups are actively using the extensibility features to provide distributed databases for new distributed applications.

7.4.2 Computer Mail Interoperation

The experiment in computer mail interoperability has proven successful with the test community. The capability to extend this one very popular service beyond the boundaries of the Internet may have significant impact on the options available to provide government contractors access to common computer mail facilities. Many contractors have previously been granted access to the Internet simply to enable them to communicate with government agencies. It may now be feasible to direct some of these contractors to use commercial computer mail systems and the Intermail interconnection service.

7.4.3 Protocol Studies

The selection of the IP and TCP protocols by the DoD as the basis for a DoD internetwork protocol standard shows the impact of the work of the DARPA community on DoD communication systems. The development of the Defense Data Network (DDN) by the Defense Communication Agency is a major use of these protocols in an operational military system. This influence is demonstrated further by mounting requests for information about IP/TCP from companies interested in producing commercial products or bidding on government contracts.

Through our participation in the Internet Working Group meetings and in technical meetings with other contractors, we have successfully influenced the development of many protocols and protocol features. Our publication in conferences, journals, and newsletters extends the impact of this work to others who design similar systems.

REFERENCES

1. Finn, G. G., *Routing and Addressing Problems in Large Metropolitan-scale Internetworks,* USC/Information Sciences Institute, RR-87-180, March 1987.
2. Lynn, J., "Privacy Enhancement for Internet Electronic Mail: Part I: Message Encipherment and Authentication Procedures," RFC 989, IAB Privacy Task Force, February 1987.
3. Reynolds, J. K., and J. Postel, "Assigned Numbers," RFC 990, USC/Information Sciences Institute, November 1986.
4. Reynolds. J. K., and J. Postel, "Official ARPA-Internet Protocols," RFC 991, USC/Information Sciences Institute, November 1986.
5. Birman, K. P., and T. A. Joseph, "On Communication Support for Fault-Tolerant Process Groups," RFC 992, Cornell, November 1986.
6. Clark, D. D., "PCMAIL: A Distributed Mail System for Personal Computers," RFC 993, Massachusetts Institute of Technology, December 1986.
7. ANSI X3S3.3, "Final Text of DIS 8473, Protocol for Providing the Connectionless-mode Network Service," RFC 994, March 1986.
8. ANSI X3S3.3, "End System to Intermediate System Routing Exchange Protocol for Use in Conjunction with ISO 8473," RFC 995, April 1986.
9. Mills, D. L., "Statistics Server," University of Delaware, RFC 996, February 1987.
10. Reynolds, J. K., and J. Postel, "Internet Numbers," RFC 997, USC/Information Sciences Institute, March 1987.
11. Clark, D. D., M. Lambert, and L. Zhang, "NETBIT: A Bulk Data Transfer Protocol," RFC 998, Massachusetts Institute of Technology, March 1987.
12. Westine, A. W., and J. Postel, "Request for Comments Summary, Notes: 900-999," RFC 999, USC/Information Sciences Institute, April 1987.
13. Reynolds, J. K., and J. Postel, "The Request for Comments Reference Guide," RFC 1000, USC/Information Sciences Institute, August 1987.
14. NetBIOS Working Group, "Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Concepts and Methods," RFC 1001, March 1987.
15. NetBIOS Working Group, "Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications," RFC 1002, March 1987.
16. Katz, A., "Issues in Defining an Equations Representation Standard," RFC 1003, USC/Information Sciences Institute, March 1987.
17. Mills, D. L., "A Distributed-Protocol Authentication Scheme," RFC 1004, University of Delaware, April 1987.
18. Khanna, A., and A. Malis, "The ARPANET-AHIP-E Host Access Protocol (Enhanced AHIP)," RFC 1005, BBN Communications Corporation, May 1987.

19. Rose, M., and D. Cass, "ISO Transport Service on Top of the TCP Version 3," RFC 1006, NRTC, May 1987.

20. McCoy, W., "Military Supplement to the ISO Transport Protocol," RFC 1007, June 1987.

21. McCoy, W., "Implementation Guide for the ISO Transport Protocol," RFC 1008, June 1987.

22. Braden, B., and J. Postel, "Requirements for Internet Gateways," RFC 1009, USC/Information Sciences Institute, June 1987.

23. Reynolds, J. K., and J. Postel, "Assigned Numbers," RFC 1010, USC/Information Sciences Institute, May 1987.

24. Reynolds, J. K., and J. Postel, "Official Internet Protocols," RFC 1011, USC/Information Sciences Institute, June 1987.

25. Reynolds, J. K., and J. Postel, "Bibliography of Request for Comments 1 through 999," RFC 1012, USC/Information Sciences Institute, June 1987.

26. Scheifler, R. W., "X Window System Protocol, Version II Alpha Update - April 1987," RFC 1013, Massachusetts Institute of Technology, June 1987.

27. "XDR: External Data Representation Standard," RFC 1014, Sun Microsystems, June 1987.

28. Leiner, B., "Implementation Plan for Interagency Research Internet," RFC 1015, RIACS, July 1987.

29. Prue, W., and J. Postel, "Something a Host Could Do with Source Quench: The Source Quench Introduced Delay (SQuID)," RFC 1016, USC/Information Sciences Institute, July 1987.

30. Leiner, B., "Network Requirements for Scientific Research," RFC 1017, RIACS, August 1987.

31. McKenzie, A., "Some Comments on SQuID," RFC 1018, BBN Labs, August 1987.

32. Arnon, D., "Report of the Workshop on Environments for Computational Mathematics," RFC 1019, Xerox Palo Alto Research Center, September 1987.

33. Romano, S., and M. Stahl, "Internet Numbers," RFC 1020, SRI International, November 1987.

34. Partridge, C., and G. Trewitt, "The High-Level Entity Management System (HEMS)," RFC 1021, Stanford, October 1987.

35. Partridge, C., and G. Trewitt, "The High-Level Entity Management Protocol (HEMP)," RFC 1022, Stanford, October 1987.

36. Partridge, C., and G. Trewitt, "HEMS Monitoring and Control Language," RFC 1023, BBN/NNSC, October 1987.

37. Partridge, C., and G. Trewitt, "HEMS Variable Definitions," RFC 1024, Stanford, October 1987.

38. Postel, J. "TCP and IP Bake Off," RFC 1025, USC/Information Sciences Institute, September 1987.

39. Kille, S. E., "Addendum to RFC 987 (Mapping between X.400 and RFC-822)," RFC 1026, University College London, September 1987.

40. Carl-Smoot, M., and J. Quarterman, "Using ARP to Implement Transparent Subnet Gateways," RFC 1027, Texas Internet Consulting, October 1987.

41. Davin, J., J. Case, M. Fedor, and M. Schoffstall, "A Simple Gateway Monitoring Protocol," RFC 1028, November 1987.

42. Lambert, M., "On Testing the NETBLT Protocol over Divers Networks," RFC 1030, Massachusetts Institute of Technology, November 1987.

43. Lazear, W., "MILNET Name Domain Transition," RFC 1031, MITRE Corporation, November 1987.

44. Stahl, M., "Domain Administrators Guide," RFC 1032, SRI International, November 1987.

45. Lottor, M., "Domain Administrators Operations Guide," RFC 1033, SRI International, November 1987.

46. Mockapetris, P., "Domain Names - Concepts and Facilities," RFC 1034, USC/Information Sciences Institute, November 1987.

47. Mockapetris, P., "Domain Names - Implementation and Specification," RFC 1035, USC/Information Sciences Institute, November 1987.

8. MULTIMEDIA CONFERENCING

Research Staff:	*Research Assistants*:	*Support Staff*:
Steve Casner	Brian Hung	Kathleen Sullivan
David Walden		

8.1 INTRODUCTION

The Multimedia Conferencing project has focused on issues in real-time, multisite conferencing over packet-switched networks. The goal of this research has been to develop a multimedia conferencing architecture and supporting communication protocols. This allows the combination of temporal media, such as voice and video, with shared computer-based data in an integrated, synchronized manner. This architecture must accommodate the vastly different transport requirements for the different types of media, yet provide common control.

8.2 PROBLEMS BEING SOLVED

The purpose of multimedia conferencing is to enhance the productivity of communications between individuals through the use of computers and networking. The DoD requirement to link all commanders with multimedia, command-controlled communication worldwide is a prominent example.

The Multimedia Conferencing project is concerned with the combination of conventional communication, such as real-time voice and video, with computer-based information which might include computer-generated data like maps, status displays, text, and spreadsheets. The main focus has been on providing multimedia protocols and support. This has led to a need for standardization of formats and interfaces to devices for different media. ISI, for example, has taken the lead in providing real-time video over packet-switched networks such as the Wideband Satellite Network (WBnet) [7]. In addition to support for packet-switched streams, such a system should consider interfaces to the phone system, video conferencing services, and other outside communications media. The ability to use different media is important, both to improve the speed and effectiveness of collaborative work, and also as a way to make computer-aided conferencing useful to people who are not computer professionals.

The conferencing system is being designed to accept and distribute information from existing applications with minimal integration effort. It should also have provisions for input and output of hardcopy text and images. For instance, mixed-media documents created during a conferencing session might be simultaneously displayed, edited, and then distributed to different force commanders according to their specific needs.

Since experts are our scarcest resource, the conferencing system must be as accessible to beginners as it is to technocrats. Therefore, attention to a user interface that addresses the needs of commanders and computer specialists alike is imperative.

The architectural model for conferencing explicitly aims at producing tools for different conference styles, for experimenting with prototypes, and for the conversion of other applications to conferencing modes of operation (i.e., tools that allow an otherwise single-user program to be shared among conference participants).

The implementation of a multimedia conferencing system must meet the military requirement for survivability through redundancy. Another consideration is how best to continue operation in the face of resource loss. The teleconferencing system should also preserve interoperability with DoD standard protocol services.

8.3 GOALS AND APPROACH

The three major areas of effort in the Multimedia Conferencing project are conferencing management protocols, media protocols, and demonstrations. Our goal is to develop and specify these protocols, then implement them in prototype conferencing systems to demonstrate the feasibility and effectiveness of multimedia conferencing.

1. **Conferencing Management Protocols.** An overall conferencing management architecture is being specified and developed in cooperation with other ARPA contractors [1, 9, 10, 13]. This task area includes concerns of general conference architecture, control protocols, synchronization, information distribution, multimedia datastream connectivity, information pre-delivery, conference entry and exit management, and user interaction.

2. **Media Protocols.** The individual media building blocks are also being specified and developed. This task area includes the design and implementation of the individual media protocols and algorithms.

3. **Demonstrations.** The prototype conferencing system is available for regular use by research groups that would otherwise have to travel for meetings. The feedback from users not involved in the implementation of the conferencing system is invaluable for learning how the system can be improved.

8.4 SCIENTIFIC PROGRESS

8.4.1 Overview

The project, a collaborative effort between ISI and BBN Laboratories, has developed an experimental system for real-time, multisite conferences. The three essential components of the conferencing environment are voice, video, and a shared workspace. The underlying communication model is based on packet-switched voice and video protocols for real-time data, operating in the DARPA Internet. Due to high bandwidth requirements, communication is currently possible only over the WBnet. A Sun workstation serves as a shared workspace, where the Diamond/MMCONF system [2, 3, 14] is used for the presentation and collaborative editing of mixed-media documents.

The teleconferencing system has been constructed from components developed by previous projects. Packet voice and packet video facilities developed under the Wideband Communications project at ISI provide isochronous communication, while the Diamond/MMCONF software was developed by the Diamond Project at BBN and runs on a Sun-3 workstation. The media components are being integrated for easy user control of a conference, while individually they are being refined for improved performance.

During a conference, the video image from the remote site is displayed on the video screen. A room-view camera captures the panoramic view of the conference participants at each site. There are remote controls for the camera, enabling it to pan from left to right, to zoom in for a closer view, or to adjust the focus. Voice data is played back through headphones so that it will not be picked up by the open microphone and echoed back to the remote site. The headphones are attached to individual infrared or radio receivers, so participants are not wired to the table. This arrangement may be replaced in the future by a loudspeaker and an acoustic echo canceller if adequate cancellation can be achieved.

Voice and video are processed by separate VT, PVP, and VF programs in a dedicated BBN Butterfly multiprocessor called the Voice Funnel (see Figure 1). The VT and PVP programs packetize the voice and video data, provide reordering and buffering of packets on receipt, etc. (see the following sections). The VF program acts as a gateway to the WBnet, allocating reserved bandwidth for voice and video traffic. It is connected as a host to a Butterfly Satellite IMP (BSAT) [7].

The video system has three components; a front-end video I/O system, an image-processing section, and a data-transmission section to connect to the Voice Funnel. The video I/O system (a commercial graphics system) consists of a camera, a 19" monochrome display monitor, and an image data buffer for images scanned in from the camera or for display. The image-processing section (composed of a processor from the commercial system and a special processor built by ISI) includes the system controller, which manages all three sections, and the video codec processor. The data-transmission section includes a high-speed data link that exchanges data between a compressed-data buffer (used by the image-processing section) and the PVP program that formats the data for the Voice Funnel. Pictures are input from the camera, compressed, and sent to the Voice Funnel for transmission. Incoming data is received from the Voice Funnel, expanded, and displayed.

Analog voice data from the microphone is converted to a 64 Kb/s PCM digital signal by the Switched Telephone Network Interface (STNI) in the Voice Funnel. Digitized voice information is forwarded by the STNI to the VT program for packetization. Likewise, incoming digital voice information is converted back to an analog signal through the STNI. Essentially, the STNI acts as a voice codec, but it does not perform any compression or decompression of data.

The live audio and video components of the conferencing system are combined with a shared, computer-based workspace for collaboration on Diamond multimedia documents. A Diamond mixed-media document may contain text, graphics, bit-mapped images, speech, and spreadsheets. Such a document may serve as the basis for the conference's technical discussions or the meeting's agenda, or for note-taking as the conference progresses. MMCONF acts as a conferencing umbrella program to Diamond (or other programs) in that it allows the Diamond editor, which is otherwise a single-user program, to be shared among conference participants. It takes keyboard and mouse inputs generated by any participant who has the conference "floor" and duplicates them at all sites. Once a Diamond mixed-media document is replicated at each site, MMCONF keeps the copies in synchronization during display and editing.

8.4.2 Real-Time Conferencing

In its youth, the prototype multimedia teleconferencing facility was strictly a two-site system. Conferences were held between meeting rooms at ISI and BBN. The WBnet carried both the video channel from ISI's packet video system and the shared-workspace display of BBN's Diamond/MMCONF multimedia conferencing system. Video traffic used the ST protocol (see Section 1.4.3, on Network Protocols) through the Voice Funnel gateway, and Diamond/MMCONF used the IP and TCP protocols through the Butterfly gateway and the ARPANET. A Shure ST-3000, more or less a glorified speakerphone, was used for the transmission of audio information.

From all appearances, the communication path provided by the teleconference facility allowed the meetings to be as effective as face-to-face meetings.

Since then, the system has changed in a number of respects. All conference traffic is now carried over the WBnet. The packet voice system now provides a true "4-wire circuit" between sites, allowing us to test different methods of conference room echo suppression. Although more work is needed on the audio system, participants generally agree that the video teleconference provides an effective and convenient means for groups to meet.

The multimedia conferencing facility has been tested not only during experimental teleconferencing sessions, but also during official meetings of task forces, technical gatherings, and other scenarios where group participants have been bi-coastally dispersed. Simultaneous document display and collaborative editing under the Diamond/MMCONF system have worked well; revisions to the ST protocol document itself, review of results from experiments of the NETBLT protocol over the WBnet, work on circuit design diagrams, all have served as the basis for successful multimedia conferences. Additionally, prepared diagrams and slides have been brought on paper and scanned as bitmap images into Diamond during meetings.

Over the course of the year, enhancements to the W net hardware and software improved the reliability of the voice and video communication subsystems, resulting in meetings with as long as nine hours of continuous operation. In May 1987, a new teleconference room at ISI was made available. It has a custom-designed, half-circle, 6-foot-diameter table with angled insets where two Sun monitors are mounted (see Figure 2). Six people can sit around the table while viewing the monitors and still remain within the field of view of the camera. We have slaved two monitors to one Sun for better viewing. The availability of a dedicated room provides more opportunities for others to make use of the teleconference system.

The participants have returned several valuable comments on how the teleconferencing system could be improved. Some new features have been added to MMCONF based on these comments. Video resolution was increased 25 percent in the horizontal dimension so that pixels now have a 1:1 aspect ratio. The packet video data rate was also increased, taking advantage of improved performance of the WBNet. The most telling comment is that participants request to use the system again and again.

8.4.3 Network Protocols

The Internet Protocol (IP) [11] and the Transmission Control Protocol (TCP) [12] are widely used in the DARPA Internet for data-oriented packet communication. However, these protocols are inappropriate for real-time packet voice and video. TCP uses an end-to-end acknowledgment and retransmission strategy to ensure reliable delivery of data, but does so by adding extra delay. For interactive voice and video, it is more important to minimize delay than to ensure reliable delivery. Therefore, an alternate suite of protocols has been developed; the Stream Protocol (ST) [8, 15], the Network Voice Protocol (NVP) [4, 5], and the Packet Video Protocol (PVP) [6]. Using this set of protocols, the conferencing system tolerates occasional packet damage or loss without severe degradation in quality.

ST operates at the same level as IP but is connection-oriented instead of datagram-oriented (see Figure 3). Features include network resource reservation facilities, multidestination addressing for multisite conferences, small packet headers to reduce delay, and aggregation of packets from multiple users into more efficient large packets for transmission across long-haul paths. On the WBnet, the ST protocol's use of reserved-stream bandwidth cuts transmission delay in half and minimizes delay variations. Unlike IP gateways, ST gateways such as VF must keep track of state information about connections established through them. Connection setup involves processing in both hosts and gateways to establish a static route according to resource requirements.

The specifications for NVP and PVP both consist of a control protocol and a data protocol. The protocol messages are supposed to be carried in IP datagrams before the ST connection is established, although the current PVP implementation does not follow this model. This setup allows the callee to accept or reject the connection and to verify the compatibility of the data-encoding algorithms. If the encoding can operate over a range of parameters, negotiation of the ST connection's network-specific resource allocation values occurs. The data protocols for NVP and PVP simply specify the headers to be included in data packets. The header fields provide sequencing and other control information as required for the particular medium.

In the current implementation of PVP (i.e., the PVP program as shown in Figure 1), no IP packet exchange precedes the opening of the ST connection as is done in the packet voice system. A substantial performance improvement has been achieved by implementing ST within the PVP program to replace the IP datagram transmission used initially. In addition to supplying the data packet headers for ST, the program is responsible for the control packet exchanges necessary for establishment and termination of the ST connection.

8.4.4 Video System

The PVP program has two I/O ports, one for the data link and another to connect to the Voice Funnel. On the transmit path, it completes the packetization of video data according to the PVP protocol, performs packet bookkeeping chores, and regulates the flow of packets to the Voice Funnel. On the receive path, it groups and reorders packets to insure that all blocks processed in one 32-block cycle belong to a single frame. The program does further consistency checks and packet bookkeeping and regulates the flow of packets to the video hardware. The PVP program functions asynchronously with the Voice Funnel but synchronizes packet flow with the video hardware, on a cycle-by-cycle basis, by exchanging control packets with the video control processor. It also provides an operator interface for establishing network connections and provides statistical displays for operational monitoring. Finally, it serves as a network host, exchanging outgoing and incoming communication packets with the network gateway, the Voice Funnel.

A quantum improvement in performance was achieved with the implementation of a "variable-frame-rate" compression and transmission scheme. This scheme incorporates interframe coding to avoid processing and transmission of any parts of the image that are unchanged from the previous image. This allows a reduction in the data rate by as much as a factor of eight, or conversely an increase in the image update rate by the same factor. The system currently implements the latter technique, producing a variable frame rate (update rate) but a constant data rate.

Originally, our teleconferences with packet video were limited to two sites because we had only two sites equipped with the conferencing system. Since other groups expressed interest in packet video, we investigated the possibility of adapting a commercial video codec to the packet network instead of building more of our own, allowing research in management of multisite conferences. This provides the option to extend the system beyond four sites, which was not practical with our experimental codec. The commercial vendors have advanced the technology considerably since we began work on our codec, but the problem that led us to seek our own solution existed for some time: the commercial systems did not tolerate packet loss well.

In order to understand how to adapt a commercial codec to be tolerant of packet loss, we studied the packet loss rate and distribution for video traffic on the WBnet. We augmented the Packet Video Protocol implementation to allow the detection of missing packets (since the experimental packet video system didn't care about missing packets), so we could run video for an extended period of time to collect loss information.

While only two copies of the video hardware existed, work began on augmenting the conferencing system to understand not only point-to-point connections between two sites, but also multipoint connections among two or more sites. Initially a "fake" video source was used for the third site. Since there were only two copies of the prototype packet video hardware, initial testing simulated three-site video by having one site connect back to itself while also connected to the second site. This allowed an interim version of the ST conferencing protocol to be developed in the Packet Video Program; BBN implemented the Voice Funnel side of the protocol. The conversion to commercial video hardware will allow more than two systems to be installed.

In September 1987 we established our first three-site video conference connection, using an interim implementation of the ST conference protocol in the PVP program and Voice Funnel, and multicast group addressing in the WBnet BSATs. The three sites were ISI, BBN, and Fort Knox (the Fort Knox site is physically located at BBN, but shares only the analog portion of the BBN earth station and is effectively the same as a separate site). Each site produced a video data stream of 230Kbps, which was delivered through the multicast mechanism to the other two sites.

Since we have only two real video-compression systems, the Fort Knox copy of the PVP program was put into a data-echo mode to return the data stream of the first participant that connected to it. At the ISI and BBN sites, the display on the video monitor was split into top and bottom halves to show the other two sites. For example, if ISI connected to Fort Knox first and then BBN connected to ISI, the display at ISI would show an echo-delayed (via Fort Knox) ISI image on top, with a direct BBN image on the bottom. At BBN, the direct ISI image would be shown on top with an echo-delayed ISI image on the bottom -- an interesting effect! We then reversed roles by re-establishing the connection and setting up the BBN-to-Fort-Knox link first.

In the future we will replace the two prototype video-compression systems with commercial hardware so that the number of real sites can be expanded. We have proceeded with the purchase of commercial codecs to replace the experimental codecs built at ISI for use in multimedia conferencing. This will allow us to expand from two packet-video sites to four over the next several months.

8.4.5 Voice System

When the packet voice system is used for teleconferencing, packet communication is the easy part. One of the most difficult aspects of meeting-style teleconferencing is the voice input and output. It is very hard to have an interface that is both comfortable and natural for the participants, while avoiding acoustic echo feedback in the room.

It's clear from teleconferences we have conducted that, even though all the media work together to make the conference effective, voice is the most crucial for conveying hard information. More work is required to make the audio system fully satisfactory. Having set up a permanent teleconference room at ISI, we now have the ability to more carefully tune the audio system. We have conducted several multimedia teleconferences using various combinations of speakerphone, microphone and speaker equipment, headphones, input sensitivities, output levels, turnaround timing in the echo suppression, and 2-wire vs. 4-wire STNI interfaces. The 4-wire STNI allows a full-duplex communication path to be established. Unfortunately, when it is used with an open microphone and speaker, acoustic echo results. This has led to experimentation with the use of headphones and an echo canceller. This is still an area of exploration.

While the packet voice facility is part of the multimedia teleconference system, it can also be used for normal point-to-point voice calls. One function of the STNI is to allow use of the packet voice system from any telephone. The STNI converts touch-tone input to ASCII characters to be parsed by the VT program. VT routes the request across the network to the destination STNI, which dials the desired number.

The use of the packet voice system with a conventional telephone also gives rise to echo problems. The echo is caused by the 2-to-4-wire hybrid circuit that is responsible for coupling the 2-wire telephone with the digital voice path. Therefore, we have begun looking into the development of an echo canceller to incorporate into the STNI. We plan to use the NEC PD7720 Signal Processing Interface (SPI) integrated circuit for this purpose.

8.4.6 Image Scanner

We developed an image scanner application on the IBM-PC/AT in order to scan offline material into multimedia documents using an attached Microtek MS-200 scanner. Later this document-scanning program was incorporated into a multimedia mail program that composes and sends messages with both text and bitmap data. Functionality akin to TFTP was added to the original version of the program to provide distribution of the multimedia messages. Messages were originally sent via the old Multimedia Message Content Protocol (MMCP), but modifications to the multimedia message format on the PC allowed such documents to be delivered to BBN's Diamond Multimedia System.

Other manipulations on scanned images that are provided by the scanner applications include the specification of contrast and brightness parameters, as well as clipping coordinates, reduction in image size, display on a high-resolution screen, reading and writing the images from or to files, and the ability to send the data to an Imagen printer (in RFC-797 format). To make the printed hardcopy the same size as the original, it is necessary to convert from the scanner resolution of 200 dots per inch to the Imagen printer's 300 dots per inch.

8.4.7 Demonstrations

In August of 1986 there was a Multimedia Mail demonstration for Dongman Lee from KAIST in Korea, and for visitors from University of Davis. A BBN Diamond Multimedia Mail demonstration was held for Peter Evans of TELECOM, Australia, in November, and for Dave Taylor of HP in December 1986. Paul Kirton of TELECOM, Australia, participated in a demonstration in March 1987. Gordon Bell of NSF and Ed Brown of DARPA were present for demonstrations in July 1987. In October 1987, participants in EDUCOM '87 at USC viewed a demonstration of the conferencing system.

8.5 IMPACT

The evolution from text-only mail to multimedia mail, to document conferencing, and finally to real-time multimedia conferencing has allowed us to explore new modes for user-to-user information exchanges. Development of an integrated multimedia conferencing system continues to broaden our expectations for how collaborative work can be achieved. Furthermore, the combination of text, speech, graphics, and facsimile into a common framework may have substantial impact on other applications as well.

We have taken the lead in developing the packet voice and video protocols for use in real-time multimedia conferencing. The requirements of packet voice and video have been the primary driving force behind the development of the high-bandwidth packet network technology represented by the Wideband Network project. This effort has demonstrated that packet-switched communication of real-time data is not only possible, but also fairly successful in the Internet environment.

Deployment of the packet teleconference system is being expanded from the existing sites at ISI (Marina del Rey, CA) and BBN (Cambridge, MA) to include new sites at the DARPA office (Washington, DC) and at SRI (Menlo Park, CA). This effort includes the installation of additional hardware, plus the development of multidestination support in the ST, PVP, and NVP implementations. This will permit the image seen by one camera to be distributed through the broadcast medium of the satellite channel to all sites participating in the conference.

A new emphasis in future multimedia conferencing work will be on the development of a standard architecture and protocols for conferencing. The purpose is to create standard building blocks and models for the conferencing infrastructure. This will avoid duplication of effort between media or contractors, and will promote an open architecture conducive to experimentation [1, 9, 10, 13].

The U.S. military has a pervasive need for timely, reliable, and effective information transfer both in times of crisis and as a part of normal operations. The multimedia conferencing system meets such requirements.

REFERENCES

1. "ANSA Reference Manual: Release 00.03 (Draft)," Advanced Networked Systems Architecture Project, Cambridge, United Kingdom, June 1987.
2. "Diamond Multimedia Document System: User's Reference Guide, Release 3.1," BBN Laboratories, Inc., November 1987.
3. "Diamond Multimedia Document System: Tutorial, Release 3," BBN Laboratories, Inc., February 1987.
4. Cohen, D., *Specifications for the Network Voice Protocol*, USC/Information Sciences Institute, RR-75-39, March 1976.

5. Cohen, D., "A Network Voice Protocol NVP-II" and "Sample NVP/ST Scenarios," USC/Information Sciences Institute, April 1981.

6. Cole, E. R., "PVP - A Packet Video Protocol," USC/Information Sciences Institute, W-Note 28, August 1981.

7. Edmond, W. B., S. Blumenthal, A. Echenique, S. Storch, T. Calderwood, and T. Rees, "The Butterfly Satellite IMP for the Wideband Packet Satellite Network," *Proceedings ACM SIGCOMM,* pp. 194-203, August 1986.

8. Forgie, J. W., "IEN 119: ST -- A Proposed Internet Stream Protocol," unpublished memorandum, MIT Lincoln Laboratory, September 1979.

9. Garcia-Luna-Aceves, J. J., E. J. Craighill, and R. Lang, "An open-systems model for computer-supported collaboration," in *Proceedings of the Second IEEE Conference on Computer Workstations,* 1988.

10. Lantz, K. A., P. P. Tanner, C. Binding, K. Huang, and A. Dwelly, "Reference models, window systems, and concurrency," *ACM SIGGRAPH - Computer Graphics,* vol. 21, no. 2, April 1987, pp. 87-97.

11. Postel, J. B., "Internet Protocol," RFC 791, USC/Information Sciences Institute, September 1981.

12. Postel, J. B., "Transmission Control Protocol," RFC 793, USC/Information Sciences Institute, September 1981.

13. Sventek, J.., "An architecture supporting multi-media integration," *IEEE Computer,* pp. 46-56, 1987.

14. Thomas, R. H., H. C. Forsdick, T. R. Crowley, R. W. Schaaf, R. S. Tomlinson, V. M. Travers, and G. G. Robertson, "Diamond: A multimedia message system built on a distributed architecture," *IEEE Computer,* December 1985, pp. 65-77.

15. Topolcic, C., and P. Park, Draft of "Proposed Changes to the Experimental Internet Stream Protocol (ST)," BBN Laboratories, April 1987.

(Note: RFC and IEN references are available from the Network Information Center, SRI International, Menlo Park, California.)

9. PROTOCOL ACCELERATOR

Research Staff:
Paul Mockapetris

Support Staff:
Kathleen Sullivan

9.1 PROBLEM BEING SOLVED

Existing internets, such as the DARPA Internet, are built with processing components and communications links of widely varying capabilities. There is a growing need to further expand the range of communication speeds, processing technologies, and other communication systems that can be incorporated in this technology. The primary examples are the promise of gigabit fiber optic media and the switching/processing power of VLSI technologies, but we also would like to integrate network services with video, voice, and other services that are not traditionally packet-switched, so that they can be managed, enhanced, and used as components of network applications.

The problem is to develop a model for communications infrastructure that preserves the intellectual capital represented by our existing software and protocol base while it harnesses the potential of new media and computation resources. The Protocol Accelerator defines an overall structure for protocol support that allocates whatever protocol processing resources are available to provide the best performance that can be achieved using the available resources. In the limit, it can construct protocol processing structures that can keep pace with the fastest media, while in typical cases we seek flexibility in using whatever resources are available in the most efficient manner, all the way down to implementations relying solely on host software.

9.2 GOALS AND APPROACH

The goals of the Protocol Accelerator project include the use of standard protocols, operation at high performance levels, and a design that has a unified representation of protocol logic, allowing a series of different implementations to be constructed semiautomatically for different performance levels.

In a typical application of this technology, a host will be interfaced to the media via an intelligent interface that implements the protocol accelerator functions; however, other configurations (such as gateways) are also of interest.

9.2.1 Standard Protocols

The use of standard protocols is desirable, since it protects interoperability with existing systems and avoids the large (and often hidden) costs of designing, developing, and maintaining a set of ad hoc, performance-oriented protocols. Rather than expending scarce expertise on the development of new, incompatible, special protocols, we seek to increase the performance of existing protocols (such as TCP) by several orders of magnitude. Quantitative changes of this magnitude provide a qualitative change in the types of applications that can be supported. Note that although this research is primarily aimed at the DARPA protocol suite, the techniques are directly applicable to the ISO and other protocol suites.

A comprehensive implementation of all TCP or similar protocols to run at gigabit rates would require a vast design effort and is beyond the state of the art at present. However, it is possible to implement selected portions of the protocols, and approach very closely the performance of a comprehensive implementation, if we identify and implement those protocol operations that make up a small percentage of the possible operations but a very large percentage of typical operations.

9.2.2 High Performance Levels

The basic goal of the protocol accelerator is to provide high-performance protocol services over high-bandwidth links. This goal can be accomplished by designing an architecture for packet processing that can transform the raw bandwidth of high-speed communication media into existing protocol services. At these speeds, parallel processing elements are required. Since our goal is to deliver performance to the application, and not simply to move protocol and operating system bottlenecks up to higher levels in the protocol stack, direct memory access to the host is required.

Our immediate goal is to attain end-to-end TCP performance in the 50-100 Mbps range over 100-200 Mbps links, although the design is amenable to faster or slower links.

9.2.3 Unified Implementation

Experience suggests that the design and maintenance of protocol software is a very complex and difficult task. It would be premature to assume that the logic needed to implement even such mature protocols as TCP has reached its ultimate state, and some emerging standards change almost daily. Thus a programmable structure, with the ability to alter the program, is a requirement for the protocol accelerator.

At the same time, we would like to be able to use the same protocol software, with accelerator interfaces of varying capability, to achieve different performance levels and costs, matching the needs of the applications and hosts.

Our goals are:

- To design a methodology for expressing protocols as independent computations linked by dependencies (much in the manner of a PERT chart).
- To program a protocol statement for TCP (and lower levels) that maximizes the available parallelism.
- To design a software tool that takes the protocol statement and a description of the available configuration, and outputs host software and programs for the parallel processing components of the protocol accelerator interface, if any.

When these tools are applied to the configuration for a high-performance host and a very capable interface, the interface programming component is large and the host software component, though still much larger, is minimized. In the case of a PC or other small host, the interface component may be empty. In the case of a gateway, certain protocols or entire protocol layers may be unnecessary.

9.3 SCIENTIFIC PROGRESS

Since the design of the interface processing elements, the protocol statements, and the configuration generator all are tightly coupled, research to date has concentrated on a first iteration of each of these components, together with a survey of existing technology that can be incorporated into the design.

Our preliminary conclusions are that the existing IP/TCP protocol suite does offer opportunities for parallel processing beyond the obvious opportunities for parallel verifications of CRCs and checksums for different protocol layers. A detailed coding for worst-case (short) TCP packets revealed that, with parallel hardware customized to CRC functions, and dual general-purpose processors for protocol tasks, TCP throughput was roughly 4 times the basic processor operation rate. For example, 1 Mip processors would yield 4 Mbps of throughput, 20 Mip processors 80 Mbps, etc. This performance improves with larger packet sizes.

Another result is that these computa ·ns can be performed almost completely as the packet is arriving on the media, eliminating latency while improving throughput. Large tradeoffs among algorithmic complexity, processing speed, memory requirements, and latency are possible.

We have a preliminary design for the processing units in the interface, which was used to develop these benchmarks. Additional refinement is necessary to take advantage of rapid change in the commercially available microprocessors and programmable logic.

9.4 IMPACT

The military has an increasingly large demand for packet communication services. While the normal needs of the military are similar to those of the research and commercial communities, the military also makes special demands on its communications services, particularly in times of crisis. This research uses an architecture, called the protocol accelerator, that is particularly effective in meeting the military's specialized requirements.

The basic goal of the protocol accelerator is to provide high-performance protocol services over high-bandwidth links. The protocol accelerator accomplishes this with a design that efficiently transforms raw bandwidth into standard services. A typical goal is end-to-end TCP performance in the 50-100 Mbps range over 100-200 Mbps links, although the design is amenable to faster or slower links. In normal times, this efficient use of high-bandwidth links can transla te into either an increased capability or a lower demand, and hence cost, for communication links. In times of crisis this efficiency reduces the communications facilities that must be supported to allow the military to complete its mission.

This improvement is achieved in the context of existing DoD-standard protocol families. It also promotes interoperability and eliminates the need for development of special, performance-oriented protocols. Rather than expending scarce expertise on the design of such new, incompatible, special protocols, we increase the performance of existing protocols (such as TCP) by factors of 10, 100, or more.

Of specific interest to the military is the inherent agility in our architecture at the lower levels: it will improve performance by speeding each packet on its way, rather than simply increasing the number of slower packets sent in parallel. This agility can be used to improve the speed of reconfiguration and the allocation of resources by precedence in the face of network damage. A design supporting multiple slow streams must wait for reconfiguration information to be processed at the slow stream rate before the other parallel streams are useful; this design can dedicate full speed to routing, reconfiguration, and other activities needed to restore network function.

Since the architecture modularizes communication services in processing elements that are logically external to the host, it facilitates network-oriented security measures. This design can also be used, together with the high performance of the protocol accelerator, to allow a general-purpose host to control the flow of high-speed channels without being involved with the actual protocol tasks.

10. SUPERCOMPUTER WORKSTATION COMMUNICATION

Research Staff:
Stephen Casner
Alan Katz

Support Staff:
Kathleen Sullivan

10.1 INTRODUCTION

The Supercomputer Workstation Communication (SWC) project has explored new techniques for using high-capacity networks to communicate between personal workstations and remote supercomputers. As part of this effort, the SWC project evaluated and tested the use of standard IP/TCP-based protocols, as well as newer remote window protocols such as the X Window System and Sun Microsystems' NeWS. The DARPA Wideband Satellite Network provided the high-capacity, long-distance communication path for many of these tests. User requirements of the scientific community that will be using these systems were also investigated.

10.2 PROBLEM BEING SOLVED

The project addressed the need for effective, high-bandwidth communication between powerful remote computers and their users. The need for continued research in this area is demonstrated by the rapid growth of the Defense Data Network (DDN) (and more recently of NSFNET), and especially by the advance of technology that allows higher speed long-haul networks.

Two major initiatives, now under way, are helping to provide significantly increased computing resources to research scientists. The first, a component of the DARPA Strategic Computing Initiative, is working to develop scalable parallel architectures in order to provide a cost-effective form of very high-performance computing. The second, the NSF Supercomputer Initiative, has created a number of supercomputer sites that are connected by a high-capacity backbone network. Each site has its own regional network, which provides access to universities and other research establishments.

With the creation of supercomputer centers and high-capacity networks providing access to them, there will soon be a large number of scientists remotely accessing computer resources in order to do research. For this to be effective, it will be necessary for them to coordinate the use of remote systems with that of local resources such as mainframe computers, workstations, smaller personal computers, and laboratory equipment. Many technical issues must be resolved for this to work effectively.

The SWC project has attempted to answer questions such as the following:

1. How well do the traditional remote access protocols work on a supercomputer and over a network such as the Wideband Network?
2. Are these protocols adequate, or should additional new protocols be developed?
3. How does one effectively split computer processing between a remote supercomputer and a local workstation? What parts of an application should be run remotely and what parts must be run locally?
4. What is the best way to edit remote files and to interactively debug programs that are running remotely?
5. Is it possible to provide a generally usable means of connecting the output of a program running on one remote machine to the input of a program running on another machine? Are existing transport protocols adequate to handle this?
6. What will the user requirements of the scientific community be? What tools and protocols must exist in order for the supercomputer/workstation environment to be useful?

10.3 GOALS AND APPROACH

The main objective of the Supercomputer Workstation Communication project was to develop new techniques to allow personal workstations to communicate with remote supercomputers (such as a Cray 2) over high-capacity networks. It is common for supercomputer centers to provide high-bandwidth access for local users over Ethernets, Hyperchannels, or other high-speed networks, but remote users are often constrained by low-data-rate links. The DARPA Wideband Satellite Network, with its 3 Mb/s channel speed, provided the combination of high capacity and long distance we needed for our tests of remote access.

Our research was divided into four areas. The first area of research was the evaluation and testing of existing remote access protocols. We verified that the file transfer protocol (FTP) and remote login protocol (Telnet) work over the Internet to the five NSF-sponsored supercomputer sites. We also tested the performance of transport protocols over the Wideband Network: the well-established TCP protocol, as part of FTP, and an experimental protocol from MIT called NETBLT. [5]

In our second area of study, we explored the possibility of an Intelligent Communication Facility that would allow users to connect the output of a program running on one machine to the input of a program running on a different machine. It should be possible to "patch together" various programs on different machines, some of which may be supercomputers and some less powerful, without having to modify the programs.

The third area involved research into supercomputer and workstation interaction. We wanted to learn how to share processing between a remote supercomputer and a workstation, how the user should interact with the systems, and how to best use the new remote window protocols. The remote window protocols we studied were MIT's X Window System [23] (X for short) and Sun Microsystems' Network extensible Window System (NeWS). [25]

As part of this effort, we developed a prototype user environment under X and wrote various application programs. One such program was an interactive Mandlebrot Set viewing program. A second application was a unique type of split editor for editing remote files from a personal workstation. This split editor runs within GNU Emacs [24] and includes remote directory viewing, FTP, and remote login capability.

In our last area of investigation, we attempted to predict the future user requirements of the scientific community that will be using workstations to access supercomputers. This effort included a survey of how scientists currently use supercomputers, a study of user interface issues, and a prediction of additional protocols and tools that will be needed for the supercomputer/workstation environment to work effectively.

One of the scientists' requirements is a means to communicate mathematical equations with one another, but this is hindered by the lack of any standard for representing equations on a computer. We have worked with other groups, including those working on NSF's EXPRES project, [7, 8] to study this area and to establish requirements for such a standard.

10.4 SCIENTIFIC PROGRESS

The progress made in the Supercomputer Workstation Communication project was in four general areas. Each of these areas will be covered separately.

10.4.1 Testing and Evaluation of Traditional and New Transport Protocols

Our hypothesis is that effective remote access to the supercomputers will require more sophisticated protocols than the standard file transfer and remote login protocols. However, testing of these protocols established a useful performance baseline. The tests verified the accessibility of the supercomputers over the Internet and the functionality of the protocol implementations on those machines.

10.4.1.1 Telnet and FTP on the NSF supercomputers

We surveyed the Internet accessibility of the five NSF-established supercomputer centers. Connections were made from a Sun 3 workstation at ISI directly to the supercomputer at each site (where possible) or to a front-end VAX (for the batch-oriented machines). Since none of the supercomputers are connected to the Wideband Satellite Network, we performed these tests over the ARPANET, which was extremely congested at times during the period of our tests (early 1987).

We tested five supercomputers: the Cray XMP running CTSS at the San Diego Supercomputer Center; the Cray XMP running CTSS at the National Center for Supercomputing Applications (NCSA) at the University of Illinois Urbana-Champaign; the IBM 3090 running VM at Cornell University; the Cray XMP running COS at the Pittsburgh Supercomputer Center; and the Cyber 205 running VSOS at the John Von Neumann Center (JVNC) in Princeton. The Cyber 205 was to be replaced by the long-awaited ETA-10, but this was not in place when we did our testing.

Telnet worked to all five sites with only the usual network delay common to the ARPANET at that time. We were able to directly log in to the two CTSS Crays. In order to use Cornell's IBM 3090, it was necessary for us to run a program locally in order to emulate an IBM 3270 intelligent terminal on the Sun. This was the only type of terminal that would work with the 3090.

The last two systems mentioned above are batch-oriented machines and were accessed by logging in to a front-end VAX running the VMS operating system. Jobs were submitted to the supercomputers via a batch queue on the front-end machine. We were able to Telnet to these front-end machines as well.

We were able to FTP files to and from the Cornell, NCSA, and JVNC computers. FTP was not available at San Diego at the time of the tests but was being developed. The Pittsburgh computer had user FTP only, which meant that it was necessary to log in to their system and run FTP in order to transfer files from/to it. This constraint would make it impossible to run a background FTP job from a workstation. [12] Again, this situation may have changed since the time of our test.

It should be mentioned here that UNIX will probably become the operating system for most supercomputers in the future. The JVNC has received its ETA-10 system, which runs UNIX. Also, the Cray sites have plans to change to Unicos, which is UNIX for the Cray.

10.4.1.2 FTP performance over the Wideband Network

In anticipation of the connection of supercomputers to the Wideband Network, we tested the performance of FTP for file transfers between smaller computers that are already part of the Wideband Network. The performance was limited not by processing power, but by the use of TC' as the transport protocol underneath FTP.

The raw bandwidth of the Wideband Network is relatively high (3 Mb/s), but so is its round-trip delay (almost 2 seconds for datagram traffic). These factors result in a large bandwidth $\times$ delay product; that is, many bits are "in flight" at once. To keep a 1 Mb/s stream flowing would require the transmitter to send 250 kilobytes of data before waiting for an acknowledgment. For TCP, this number corresponds to the "window size" which, in a typical implementation, is only 2 kilobytes. If the transmitter can send only 2 kilobytes and must then wait 2 seconds for the acknowledgment to be returned from the receiver, the resulting throughput is only 16 Kb/s. Experimental results demonstrated this limitation clearly.

The TCP window size of 2 kilobytes was chosen to avoid overflowing the buffer capacity of gateways when trying to send data through the lower-bandwidth ARPANET. The protocol field carrying the window size is 16 bits, so a window size up to 65,535 bytes could be used without modifying the TCP protocol. Our tests were performed between a Sun workstation at ISI and a VAX at BBN. We were able to adjust the TCP window size in the Sun by patching kernel variables, and we were able to adjust the window size per connection in the BBN VAX TCP implementation through a special command added to the FTP user program.

We found that throughput increased linearly as we increased the window size. At 15 kilobytes, the transfer rate was approximately 60 Kb/s. We were unable to increase the window size to 16 kilobytes or above because of a limitation in the arithmetic of the TCP implementation on the Sun at that time (SunOS 2.0).

To test the capacity of the Wideband Network without acknowledgment delays or window-size limits, we began a separate set of tests using the ICMP Echo Request/Reply protocol as implemented by the "ping" program on a Sun workstation. The ping program transmits a sequence of echo request packets and keeps statistics on the replies that are returned. We were able to transmit 500 Kb/s from ISI to BBN and back to ISI, for a combined 1 Mb/s on the Wideband Network. The packets were 1400 bytes long and were transmitted every 22 milliseconds. In a typical test, the average round-trip time was 1.9 seconds, with a maximum of 2.6 seconds. There was a 0.6 percent packet loss rate on the Wideband Network, and another 0.1 percent of the packets incurred uncorrected bit errors. These tests established the ability of the Wideband Network to carry 1 Mb/s without excessive packet loss and while maintaining expected delay times. Therefore, it seems reasonable to conclude that an FTP implementation that allowed the full 65,535-byte TCP window size would be able to achieve the calculated 250 Kb/s transfer rate.

Recently, extensions to the TCP protocol have been proposed to provide efficient operation over a path with a high bandwidth $\times$ delay product. [16] The window-size limit would be circumvented by negotiating an implicit scale factor to multiply times the window-size value carried in the header field. To reduce the effects of packet loss, another proposed extension would allow selective acknowledgment, so that the receiver could inform the sender about all segments that have arrived successfully; then only the segments actually lost would have to be retransmitted.

The indiscriminate use of large window sizes could cause severe congestion on paths with insufficient capacity. To operate efficiently over a wide range of network performance characteristics, it is necessary for the TCP window size to be adjusted dynamically. A study by Van Jacobson has shown a collection of techniques, including window size adjustment, to avoid congestion. [17]

10.4.1.3 Tests of NETBLT over the Wideband Network

The NETBLT protocol [5] developed at MIT is a promising alternative that avoids the window-size limitations of TCP. NETBLT is intended for bulk data transfer operations. In cooperation with MIT, we ran several series of tests of NETBLT transfers over the Wideband Network.

In the first series, IBM-PC/ATs were used at ISI and MIT. These machines were located on Ethernet LANs connected by Butterfly gateways to the Wideband Network. The NETBLT traffic was monitored on the Ethernet with the SpyTool program, which was developed at ISI for use in the Xerox Development Environment on a Dandelion Workstation. [10] Throughput was limited by packet loss at the receiving IBM-PC/AT. Bunching of the packets as they flowed through the network caused shorter interpacket intervals than the network interface on the IBM-PC/AT could tolerate.

To avoid the packet loss problem, the NETBLT implementation was ported at ISI to a Sun workstation. The Sun provides increased processing speed and a faster network interface. After a number of tests in which network timing was analyzed and NETBLT parameters were refined, we were able to achieve transfer rates near the calculated total throughput available to user traffic on the Wideband Network. The overall rate (which includes initial and final handshaking) for a transfer of 716,000 bytes was 521 Kb/s. However, the steady-state transfer rate (once the connection was established) was 942 Kb/s. This compares favorably with the theoretical maximum rate of 1.05 Mb/s calculated from the chosen NETBLT rate-control parameters. A small amount of packet loss at the destination Sun seemed to be the reason for the difference. (These results are reported by Mark Lambert in his report on tests of NETBLT at MIT. [20])

Additional tests were run betweens two Suns, one at BBN and one at ISI, with similar results.

NETBLT performance on the Wideband Network might be improved by operating it over the stream-oriented ST protocol [14, 26] rather than the datagram IP protocol. We have participated in the development and testing of ST for transmission of packet voice and video in the Multimedia Conferencing project. The bandwidth-reservation feature of ST allows for a smooth flow of data at a constant rate across the Wideband Network. This feature would mesh well with the requirements of NETBLT for rate-based flow control. Future work in this area might include the development of an FTP application program using the NETBLT protocol on top of ST.

10.4.2 Intelligent Communication Facility Study

We did a preliminary study of a system called an Intelligent Communication Facility, which would allow users to connect the output of a program running on one machine to the input of a program running on a different machine. It would be very valuable if programs on different machines, some of which may be supercomputers and some less powerful, could be linked in this way without the need for modifying the programs.

One way to accomplish this might be to extend the idea of UNIX pipes to contain typed data. One could think of these ***typed pipes*** as carrying a stream of one type of data from a process on one machine to another process on a different machine. Since the connection would be like a UNIX pipe, there would be no direct indication of how much data would be coming down it. Therefore, in order to make this type of connection work with remote machines, a remote execution protocol similar to UNIX's rsh must also exist. (We have found that such a protocol is necessary for many other applications and will discuss this further in Section 4.3.2 of this report).

In order to comply with international standards, we suggest that the data in these pipes conform to the CCITT X.409 data representation standard. [4] (See also "A Survey of Data Representation Standards." [11]) The X.409 standard originally did not include a representation for floating-point numbers, essential for scientific applications, but it has since been modified to include them. The pipes themselves could be implemented on top of TCP.

In a typical application, there may be a program that does a large amount of computation (say, running on a supercomputer) whose output consists of many pages of numbers. Another program on a second machine might accept numerical input and produce graphical data, which the user would like to display on yet a third machine, a workstation. These programs may have been written by different people. The sources to them may be unavailable. Therefore, it is desirable to have a way for these programs to communicate with one another under the direction of a user at one of the machines, without the programs themselves having to be modified in any way.

We suggest that a library of ***adapters*** should be created. Each adapter would be a small program that would read in data in a given format and then output an X.409 stream. Suppose, for example, that the first program's output consisted of a table with each line consisting of six floating-point numbers (say, in FORTRAN format 6F10.6). The adapter would read in such a line, ignore any header or title information, and output a list of X.409 floating-point numbers. On the receiving side, the second program would have an inverse adapter that would read in the X.409 list of floating-point numbers and output the equivalent data in FORTRAN format. These types of connections would continue to the third machine, the workstation (see Figure 1).

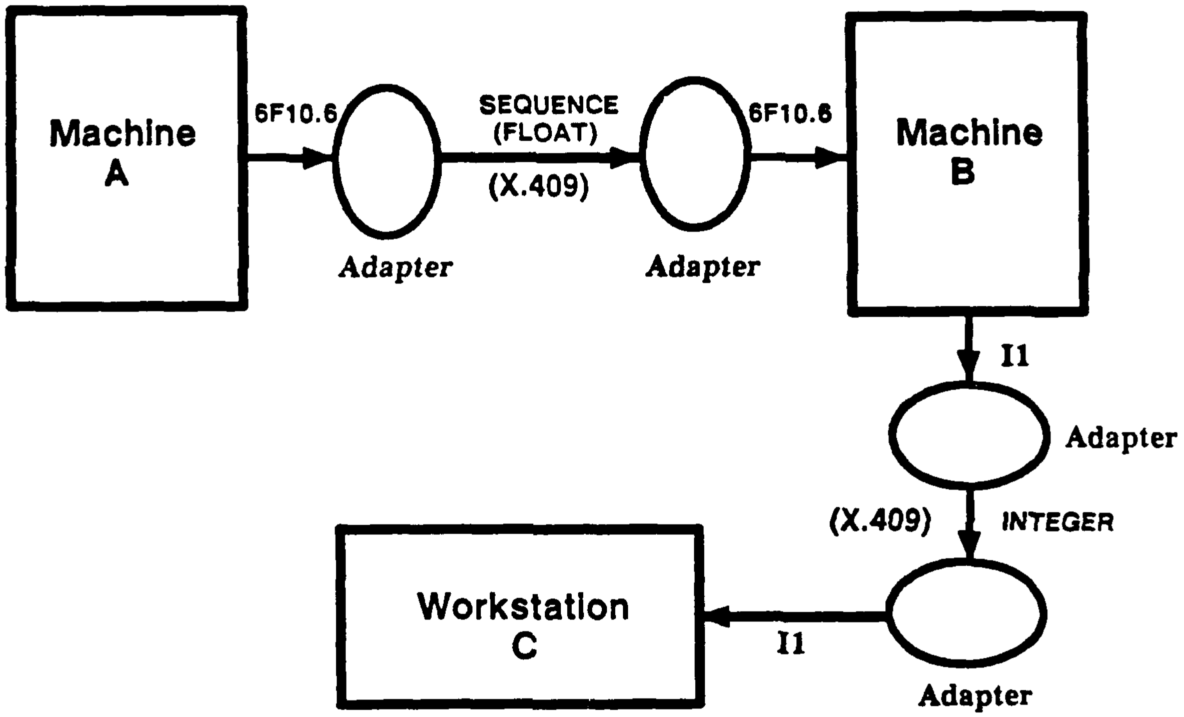


Figure 10-1: *Interconnection of typed pipes across machines*

The connections to the adapters, as well as the network typed-pipe connections, could be specified by the user logged in to any of the three machines. This could be as simple as the way it is done in UNIX (typing the names of the programs to be executed, with special characters to denote connections to pipes), or it could be much more sophisticated. One possibility is a graphical interface on the workstation that would allow a user to browse a database of programs and adapters and to specify the interconnections. Something like this was studied by Brown. [3]

If a program were written to allow the input of multiple data sets, the above system could allow for multiple pipes of input from different processes to be connected to it. Also, this type of system could work well with the remote window protocols described in the next section, allowing the older non-interactive programs to be used with a front-end that knew about the MIT X Window protocol or Sun's NeWS. The output of one such program could be in a format like PostScript on top of the X.409 connection. The user could then connect this pipe either to a display process on a workstation or directly to a PostScript printer.

We feel the constraint that the programs need not be modified is important. A general-purpose adapter capable enough to figure out the format of any type of output data is probably beyond current capabilities. However, in most cases, a user would know what type of data a program generates and could select a standard adapter or write his own. More work in this area is required.

10.4.3 Supercomputer/Workstation Interaction and Remote Window Protocols

The bulk of our project's research effort has been in the area of supercomputer/workstation interaction. This involves understanding how to split processing between the remote supercomputer and the local workstation. It is desirable for a user to be able to do this without having to devise a new communications protocol between local and remote processes for each new application.

The new remote window protocols such as the MIT X Window System allow programs running on a remote computer to display windows and to interact with the user on a local workstation in a machine-independent way. This is one natural way to split the processing between the workstation and a supercomputer. We would like to understand when it is useful to do this and when it may be better to have more of the computation occur on the workstation.

10.4.3.1 The spectrum of supercomputer/workstation interaction

In order to gain a clearer idea of this concept, it is important to keep in mind what a user needs in order to run remote programs. First, there is a need for some method of editing the programs and creating the data that will run remotely. Next, there must be some way of remotely executing these programs. The user will probably need some type of interactive debugger; in addition, since supercomputers generally use a pipelined architecture, an interactive method of measuring performance and of vectorizing code may be necessary. Finally, there must be some method of displaying the results of the computation. The user also might need to interactively modify the course of the remote computation based on a current display of its progress (for example, in a large simulation).

It is useful to consider a spectrum of supercomputer/workstation interaction (see Figure 2). At one end of the spectrum (leftmost on the diagram), as much processing as possible is done on the remote supercomputer, and a minimal amount is done locally. At the other end (at the right), the supercomputer generates only data, and all display and interaction with the user is done by the workstation. Between these two extremes, interaction with the user is shared between the workstation and the remote supercomputer. Below the spectrum line in the figure, some typical types of applications are located in their appropriate places in the spectrum.

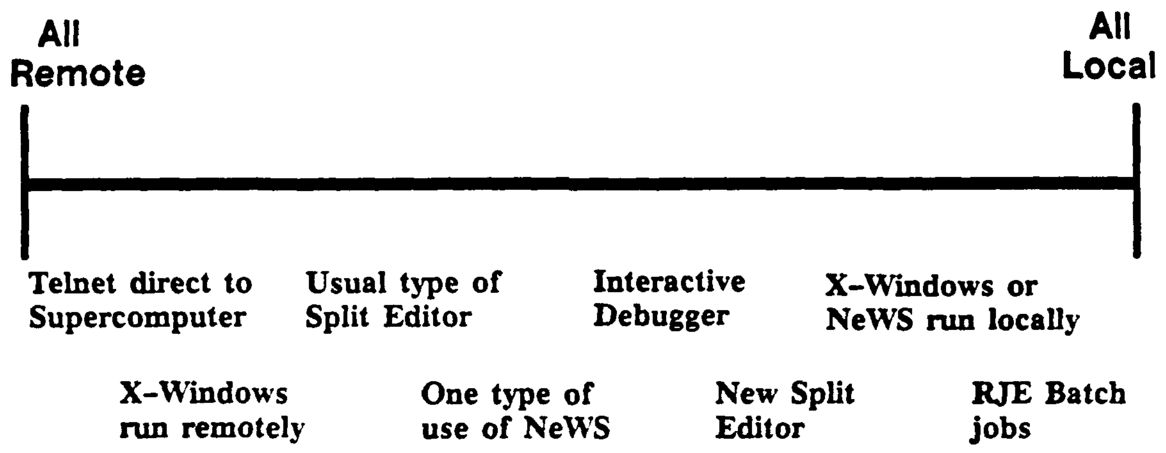


Figure 10-2: The spectrum of supercomputer/workstation interaction

We should first define the two ends of this spectrum more clearly. In order to do this, let us distinguish between the interactive part of an application (displaying graphical data in a window, interacting with the user via mouse and keyboard, etc.) and the non-interactive part. The non-interactive pieces can be run in a batch mode; that is, they (possibly) read in an input data set, do some (possibly a very large amount of) computation, and create an output data set. The user does not need to interact with this part of the program in order for it to complete its computation.

In order to take advantage of the high processing speed of the supercomputer, the user will want to run at least the non-interactive portion of an application remotely. Also, supercomputer-specific parts (such as computations that take advantage of the pipelined architecture and the execution part of a vectorizer and debugger) must be run on the supercomputer. This defines the all-local edge of the spectrum. (If we were to go any further, we would not use the supercomputer at all!)

Direct interaction with the user (paying attention to mouse clicks and characters typed) must be done on the workstation. For example, most if not all of an interactive editor should probably run on the workstation, not the supercomputer. In addition, any display of graphical information must be done locally. This defines the all-remote edge of the spectrum. Telneting directly to the supercomputer would use the workstation only to interact with the keyboard and to display text. Since this mode of operation would not allow the display of graphical information and would not take advantage of the workstation's capabilities, we will not consider it further.

A more useful approach, near the all-remote edge of the spectrum, is to use a remote window system such as the MIT X Window System [23] or Sun's NeWS. [25] In this type of system, the workstation acts as a server to process keyboard and mouse activity and to display bitmap data, but all other computation occurs in the remote computer. Because the specification of a display in the NeWS environment is a PostScript program, one possible use of this system would be to download some of the display processing to the workstation as PostScript (this is the "One type of use of NeWS" in the middle of Figure 2).

The usual types of ***split editors***, which allow the editing of remote files on a local computer, are also shown in Figure 2. As an experiment, we have developed a split editor (described in Section 10.4.3.4) that is located more toward the local end of the spectrum.

Generally, if one chooses to work in the middle of the interaction spectrum, it will be necessary to define a communications protocol to specify how the remote and local processes will interact. A future area of research would be to determine how to allow users to do this without having to completely design such a protocol. NeWS provides some of this capability (via PostScript).

At the all-local edge of the spectrum, a user may edit a program entirely at the workstation, submit it as a remote batch job to the supercomputer, transfer the output data back to the workstation, and do all display and interaction with the data at the workstation. For many applications, this is a useful mode of operation. Another project at ISI has developed an interface to FTP that works in the background to support this mode of operation. [12]

This batch-oriented style requires a generally available, IP-based Remote Job Entry (RJE) protocol. Although a general RJE protocol does not yet exist within the Internet, there are systems that work within certain segments of the Internet community. For example, an RJE system is availible on BITNET, but one must be a BITNET host to use it. The National Center for Atmospheric Research (NCAR) has developed a very nice Internet RJE system based on FTP [2] for providing remote access to their supercomputers.

10.4.3.2 Remote window protocols, the X Window System, and NeWS

Originally, we proposed to develop both a Bitmap-Telnet protocol and a remote window protocol as part of this project. However, by the time the project was started, two such protocols had been developed and are now becoming standards (MIT's X Window system and Sun's NeWS). We therefore felt it was inappropriate to create yet another protocol, and we concentrated instead on how to best use X or NeWS with supercomputers.

Both X and NeWS use a client-server model of window interaction. The user's workstation is viewed as a resource (a bitmap display, mouse, and keyboard) that is allocated by a server program running on the workstation. The server processes requests from client application programs, which may be running either on the workstation or remotely on another machine.

The major technical difference between the two protocols is that the display in a window in X is a raw bitmap and the display in a NeWS window is in PostScript format (a printer description language). Thus, a NeWS display is both hardware and resolution independent. An X display is hardware independent, but resolution dependent.

X currently seems to be in far wider use than NeWS, probably because it is available essentially free from MIT, while NeWS is a commercial product. However, Sun's merged X/NeWS window package is planned to be a standard part of UNIX, so NeWS may catch on more in the future.

X will be offered as a part of the Unicos Cray operating system and also should be a part of the next release of IBM's VM, the operating system of Cornell's supercomputer. Thus, we expect that many supercomputer users will use at least X, if not both X and NeWS.

Most current and proposed applications that use X or NeWS use these systems because they are rapidly becoming a window protocol standard, not because of their remote capability. Additional protocols are needed for these systems to be used remotely.

Primarily, a remote execution protocol is necessary. If a user on a workstation wants to access a remote supercomputer with X, something must create a process on the remote system; only then can X display this process on the workstation and allow the user to interact with it. The rsh facility in UNIX works as a remote execution protocol, but it has problems with security and authentication. Also, both the workstation and the remote computer must be running UNIX. There is a real need for a general remote execution protocol that will accept a user's password and start up a program (similar to the way FTP works).

Remote applications must be designed carefully in order to minimize their required network bandwidth. It is quite possible in either of these systems to create a large number of packets in a simple application.

10.4.3.3 Experimentation and testing with X

Some experimentation and testing was done with the X Window System. All work was done with Version 10, Release 4, on a Sun 3 workstation. The standard version of the X protocol, Version 11, has since been released by MIT.

We began with a small demonstration program, running on a VAX over our 10 Mb/s Ethernet, which continually drew moving balls in a Sun window. Using the ISI SpyTool described in Section 10.4.1.3, we found that 50 packets per second were sent to the Sun and the same number were sent back to the VAX (presumably acknowledgments). The exact same number was generated running multiple programs in different windows. This indicates that the local X server on the Sun was the limiting factor and that this should be typical of the maximum traffic one workstation might generate using X remotely. In actual practice, remote applications should be designed to exchange far fewer packets.

In addition to this testing, we built a number of applications under X. We created a package of small files that would allow a naive user of X to try it. This package, called MyX, creates an environment similar to Suntools (the Sun window package) as far as mouse interaction and window management is concerned. MyX provides users a way to try the X environment without having to wade through a large amount of documentation. We distributed MyX to a number of people throughout the Internet.

We developed an X-based graphics interface for GNU Emacs.[1] Our graphics extension allows LISP functions within this editor to create simple line-drawing graphics in an X window on the user's workstation.

We also created a larger application under X: a system for interactively viewing Mandlebrot set fractals. It allows the user to "zoom in" on a smaller region of the display. The Mandlebrot set data is calculated as it is displayed. This program runs on a remote VAX and is written in C using the Xlib interface to X. [15] If X had been available on a supercomputer, it would have been interesting to run this program on

[1]At the time of this development, GNU Emacs was one of the few editors that were able to use the X protocol.

one, as examining small areas of the Mandlebrot set can take quite a large amount of computation.

This application is a example of something near the all-remote edge of the supercomputer/workstation interaction spectrum. We would have liked to test a version closer to the other end of that spectrum, but we did not have the time to do so. Such a program might have the remote computer generate a large amount of data into a file; then a local display program would access parts of that file depending on the view the user selected.

Most of the experimental programs we developed under X were done without the use of a toolkit. However, we did write some small programs with DEC's Xt toolkit. [9] In Version 10 of X, a toolkit was a convenience but was not necessary; however, in version 11, use of some kind of toolkit is almost essential. Selection of a toolkit and a window manager (a special client program, run locally, which allows resizing and movement of windows) is a major decision a program developer must make; it will have great influence on the application's development.

We found that using a toolkit allows programming at a somewhat higher level and that programs can generally be made much shorter (as demonstrated by Rosenthal [22]). However, the existing toolkits still require programming at a relatively low level and seem more directed at systems programmers then at general application developers. There is a need for a somewhat higher level toolkit on top of the existing ones. (Note: as things are changing very rapidly, this level of toolkit may already be available. CMU's Andrew toolkit, which we did not have a chance to evaluate due to time constraints, appears to be the type of toolkit we describe.)

10.4.3.4 A new kind of remote split editor running within GNU Emacs

After considering the various types of applications on the supercomputer/workstation interaction spectrum of Figure 2, we felt it would be useful to prototype an application that falls nearer to the middle. We chose the problem of editing remote files as an example.

One usual way to edit a remote file is to FTP the file to the local workstation, edit it, and then FTP it back to the remote site. Another is to run an editor remotely on top of an interactive network connection (e.g., Telnet). Now, a user can take advantage of X or NeWS to run the editor remotely and display on the local workstation.

Both of the above approaches have potential problems. Editing a program, running it remotely, changing it, and then running it again involves transferring the entire file back and forth across the network, wasting network bandwidth. On the other hand, round-trip network delay time can make interactive remote editing very difficult.

One solution to these problems is a ***split editor***, which divides the editing computation between the local and remote machines. (An example of this is the SED editor, developed for MFEnet users. [21] Another example, closer in design to our prototype, is described by Comer, et al. [6]) Typically, a split editor works by sending one section of the file at a time to the remote machine. As the file is modified, the new versions of these sections are sent back to the remote machine. The user may run into trouble, however, when moving to a part of the file that has not yet been loaded into the local machine. When this happens, the user suddenly must wait until that section is transferred, which can be very frustrating when response has previously been very fast.

In our prototype editor, we assumed that, primarily, the user would be continually making changes to an existing remote file. If the user is creating a large file from scratch, he or she would be more likely to use a standard editor to create the file locally and then transfer the entire file to the remote machine. In our editor, we transfer the entire remote file to the local machine only once at the beginning of the editing session. From then on, we send only the changes that are made to it. The remote side of the editor makes these changes as it receives them and then acknowledges that it has done so to the local editing process. (A similar approach is also taken in a system at Purdue University. [6])

GNU Emacs [24] is an extensible editor that runs its own LISP interpreter. Editing functions are programmed in LISP and can be bound to any key. Instead of writing an editor entirely from scratch, we decided to build the split editor on top of GNU Emacs. The entire system is written in GNU Emacs LISP and requires no modification of the compiled C code that underlies the LISP. Use of the split editor appears virtually identical to the standard way in which Emacs is used.

When the user selects a remote file to edit, the entire file is transferred to the local machine and the remote side of the editor is started. Nothing is transferred to the remote side until the user makes an actual modification to the file. Then this change is sent to the remote process, which acknowledges it. The remote version of the file is saved periodically in case either machine goes down.

Changes to the file are simply GNU Emacs LISP instructions. If these instructions were saved in a file, and that LISP file were loaded into Emacs, the resulting instructions would convert the old version of the edited file to the new version. Thus, it is also possible to keep a running log of changes and to execute them all at once instead of incrementally.

Our split editor worked well. However, as was the case with the remote window protocols, a remote execution procedure was again necessary. For this split editor, we used the rsh facility of UNIX. Thus, the system as currently implemented requires both the local and remote sites to run UNIX and to have rsh access. Again, this points out the need for a general remote execution protocol not tied to any particular operating system.

As mentioned in Section 10.4.3.1, when one has an application in the middle of the interaction spectrum one generally must define a protocol for how the remote and local sides will communicate. In the case of our split editor, our protocol is just LISP instructions. This was done for ease of debugging and for readability of the changes, and because it allows for a very simple remote side. The remote process is simply another GNU Emacs process that reads in LISP instructions and executes them.

In addition to its basic editing functions, the split editor allows the user to view remote files without modifying them and to manipulate remote directories in a way similar to Emacs' Dired mode. [24]

10.4.4 Future User Requirements of the Scientific Community

In order to maximize the usefulness of supercomputers to the scientific community, it is important to determine the ways in which this community expects to use them. We therefore investigated how researchers currently use supercomputers, how they expect to interact with them in the future, and what protocols and capabilities will be necessary to support this.

We participated in the DARPA Internet Task Force on Scientific Requirements to explore these issues and to make known our results. In addition to releasing a variety of position papers on future scientific requirements for networking, the task force contributed to the 1987 FCCSET report to Congress. [13]

We interviewed several scientists who use supercomputers in order to learn what facilities currently exist and what new ones are desired. We also participated in supercomputer workshops held at NSF-sponsored supercomputer centers.

Because of the importance of mathematical equations in today's scientific communications and the lack of any standard for representing equations on a computer, we also studied the issues involved in defining an equations representation standard. Some user-interface considerations were also studied. The results were reported by Katz [18] and in a soon-to-be-released position paper by the Internet Task Force on Scientific Requirements. [19] We also participated in discussions and meetings with other groups interested in this subject, including those funded under the NSF EXPRES project (some of which was reported by Arnon [1]).

We believe there is a need for a standard that will allow for the interchange of equations between electronic mail messages, various document preparation systems, and symbolic mathematics systems. We have identified four levels of abstraction at which mathematical expressions may be represented. At the present time, there is general disagreement about which level or levels would be most appropriate for a standard. It is probable that more than one of these levels will be required. However, we believe the development of such a standard will be critical for effective computer-based communication in the sciences.

10.5 IMPACT

As a result of the NSF Supercomputer Initiative and the DARPA Strategic Computing Initiative, and because of continued increases in network and computer capability, the bulk of the scientific community will soon be using computers interconnected via the Internet in order to do their work. Many will use remote supercomputers; others will use these resources mainly to communicate electronically with their colleagues. As a result of our work, we hope they will be able to work in this new environment more effectively.

Our testing of new and existing protocols will help to determine which protocols should be used in the supercomputer/workstation environment. The development of a system similar to our proposed Intelligent Communication Facility will allow users to run existing programs on different machines without having to rewrite them or to become knowledgeable about low-level network protocols.

The current trend away from the use of large, timeshared mainframes and toward the use of powerful workstations connected to high-speed networks indicates that more and more researchers will use the new remote window protocols such as X and NeWS. Our studies in this area have given examples of how to best use these protocols and have identified several major issues that must be resolved in order for them to work.

The spectrum of supercomputer/workstation interaction provides a method for classifying different modes of using remote computing resources.

It is hoped that our work on an equations representation standard will influence the much-needed work in this area. Such a standard will be important to scientists wishing to collaborate electronically. Through our participation in various workshops and in the Internet Task Force on Scientific Requirements, we have worked toward the development of such collaborative scientific work.

10.6 FUTURE WORK

Although the Supercomputer Workstation Communication project is at an end, there is ample opportunity for follow-on work. Continuing research is needed in the following areas:

1. A general Internet remote execution protocol must be developed. This protocol should allow for user authentication and for security. Such a protocol is needed for the remote window protocols to be used effectively, as well as for applications such as our split editor and the Intelligent Communication Facility.
2. An Internet-based batch protocol is also needed. The National Center for Atmospheric Research (NCAR) has developed a such a system on top of existing protocols, [2] but this system requires the user to register a password for his local machine with the batch processor. A separate protocol is really needed, perhaps as a part of a general remote execution protocol.
3. Much more work should be done to test and tune both X and NeWS on supercomputers over high-capacity networks. At the time of our study, neither X nor NeWS was available on any of the NSF-sponsored supercomputers, but they should become available in the near future. It is unknown how much the widespread use of these protocols will load the supercomputers and the networks.
4. A standard must be defined to allow for the interchange of equations among electronic mail messages, various document preparation systems, and symbolic mathematics systems. As more and more scientists become connected to the Internet, they will demand the ability to send mathematical expressions via electronic mail. [18, 19] It would be useful to prototype the Intelligent Communication Facility described in Section 10.4.2. Having such a system in addition to X or NeWS would allow more effective use of both remote and local resources.

REFERENCES

1. Arnon, D., "Report of the workshop on environments for computational mathematics," in *SIGGRAPH Conference*, ACM, Anaheim, California, July 1987.

2. Bassett, B., "NCAR Internet remote job entry system," in *Fourth Annual Workshop on Networking and Supercomputers*, Boulder, Colorado, June 1988.

3. Brown, R. L., *A Distributed Program Composition System*, Ph.D. thesis, Purdue University, May 1988.

4. CCITT, *Message Handling Systems: Presentation Transfer Syntax and Notation, Recommendation X.409, Document AP VIII-66-E,* International Telegraph and Telephone Consultative Committee (CCITT), Malaga-Torremolinos, 1984.

5. Clark, D., M. Lambert, and L. Zhang, *NETBLT: A Bulk Data Transfer Protocol*, MIT Laboratory for Computer Science, RFC 998, March 1987.

6. Comer, D., J. Griffioen, and R. Yavatkar, "Shadow editing: A distributed service for supercomputer access," in *Proceedings of the 8th International Conference on Distributed Computer Systems*, The Computer Society and IEEE, San Jose, California, June 1988.

7. CSNET, "NSF initiates EXPRES program," *CSNET News* 12, Winter 1986.

8. CSNET, "EXPRES Project completes first year," *CSNET News* 16, Winter 1988.

9. Digital Equipment Corporation, *The Xt Toolkit*, 1987.

10. DeSchon, A., "USC Information Sciences Institute, Xerox University grant program activities," in *Proceedings of the Xerox College and University Computer Research Communications Meeting*, Xerox Corporation, Leesburg, Virginia, November 1986.

11. DeSchon, A., *A Survey of Data Representation Standards*, USC/Information Sciences Institute, RFC 971, January 1986.

12. DeSchon, A., and R. Braden, *Background File Transfer Program (BFTP)*, USC/Information Sciences Institute, RFC 1068, August 1988.

13. Federal Coordinating Council on Science, Engineering, and Technology, *A Report to the Congress on Computer Networks to Support Research in the United States*, FCCSET, Technical Report, June 1987.

14. Forgie, J. W., *ST - A Proposed Internet Stream Protocol*, MIT Lincoln Laboratory, IEN 119, September 1979.

15. Gettys, J., R. Newman, and T. Fera, *Xlib-C Language X Interface, Protocol Version 10*, MIT Project Athena, Technical Report, November 1986.

16. Jacobson, V., and R. Braden, *TCP Extensions for Long-Delay Paths*, USC/Information Sciences Institute, RFC 1072, October 1988.

17. Jacobson, V., "Congestion avoidance and control," *Computer Communications Review* 18, (4), August 1988. (SIGCOMM '88 Symposium.)

18. Katz, A., *Issues in Defining an Equations Representation Standard*, USC/Information Sciences Institute, RFC 1003, March 1987. Also published in *ACM SIGSAM Bulletin*, May 1987.

19. Katz, A., *Towards a Standard for the Representation of Mathematical Expressions*, Internet Task Force on Scientific Requirements, position paper, 1989. (To be released.)

20. Lambert, M., *On Testing the NETBLT Protocol over Diverse Networks*, MIT Laboratory for Computer Science, RFC 1030, November 1987.

21. Maron, N., "Using a personal computer as a workstation extension to the Cray/CTSS mainframe environment," in *Proceedings of a Workshop on Supercomputing Environments*, NASA Ames Research Centler, Moffett Field, California, June 1986.

22. Rosenthal, D., *A Simple X.11 Client Program, or How Hard Can it Really Be to Write 'Hello, World'?*, Sun Microsystems, Inc., Technical Report, October 1987.

23. Scheifler, R., and J. Gettys, "The X Window System," *ACM Transactions on Graphics, Special Issue on User Interface Software*, (63), 1986.

24. Stallman, R., *GNU Emacs Manual*, Free Software Foundation, March 1987.

25. Sun Microsystems, Inc., *NeWS Preliminary Technical Overview*, October 1986.

26. Topolcic, C., and P. Park, *Proposed Changes to the Experimental Internet Stream Protocol (ST),* Bolt Beranek and Newman, Inc., Technical Report, April 1987.

11. EMPIRICALLY VALID KNOWLEDGE REPRESENTATION

Research Staff:
Robert Mac Gregor
Dave Brill

Support Staff:
Ana Dominguez

11.1 PROBLEM BEING SOLVED

A dominant theme of much current AI research is the importance of explicit, well-founded models of knowledge. In the past, nearly every project with a need for a knowledge base has spent some time generating its own knowledge representation system. This has led to duplicated efforts, diversions caused by the construction and debugging of the knowledge representation, and idiosyncratic knowledge bases that cannot be shared or reused. A general-purpose knowledge representation system can have as dramatic an impact on the knowledge processing community as formalized databases have had on the information processing community. Thus, a primary goal of the Knowledge Representation project at ISI is to develop a state-of-the-art knowledge representation system, together with a set of tools that aid the process of constructing large knowledge bases. The system produced will be validated through application in a wide range of projects, including intelligent interfaces, expert systems, and natural language.

When (and only when) a useful, application-independent knowledge representation system becomes available, one can contemplate the construction of knowledge bases whose lifetimes extend beyond that of their original applications. We will need to find ways to reuse knowledge bases, and ways to share a knowledge base across multiple applications. The problem here is to develop methods for constructing knowledge bases that are accessible, extensible, and easy to debug and maintain. Once tools are available that reduce the cost of building a knowledge base, and when the cost of building a knowledge base can be amortized across many applications, we can expect that it will be commonplace for AI systems to manage much larger and more complex knowledge bases than those in use today.

The utility of a knowledge representation system is derived from four attributes.

- **It must be expressive.** The modeling tool must be able to capture the semantics of the domain in a concise, well-defined, and easy-to-use notation.
- **It must be principled.** The modeling system must have well-defined semantics so that general-purpose reasoning agents can be supplied to manage the knowledge base as it grows and to provide useful, domain-independent inferences.

- **It must be competent.** The modeling system must have reasoning facilities and an appropriate set of definition and analysis tools that cooperate with the modeler to lessen the task. The architecture must be able to manage large-scale knowledge bases.
- **It must be available.** The knowledge representation system must be robust, efficient, and well documented. The language and environment that it provides must interface easily with application programs.

While researchers have *experimented* with various methodologies and produced *experimental* languages and environments, no one has yet produced a knowledge representation system that scores well with respect to the criteria mentioned above. The goal of this project is to continue to expand the capabilities of an existing knowledge representation system, Loom [5], to fill this need.

11.2 GOALS AND APPROACH

Our work in knowledge representation is motivated by a desire to build a principled knowledge representation system, Loom, that can be used to provide representational competence in a variety of applications. To this end, we have solicited use of the Loom system in the following application areas: expert systems, intelligent user interfaces, and natural language processing. Our research methodology is to allow application needs, rather than theoretical interests, to drive the continued development of the language. This methodology has allowed us to perform an empirical evaluation of the strengths and weaknesses of Loom. It has also helped us identify general requirements for reasoning in intelligent systems.

We classify the improvements that we have made or plan to make into three broad categories:

1. **Expressiveness:** enhancements to the definitional and assertional components of Loom and to its ability to infer subsumption, implication, and instantiation relationships.
2. **Environment:** enhancements to the tools that accompany Loom for maintaining multiple knowledge bases, for modifying and extending knowledge bases, and for performing validation of knowledge.
3. **Extensibility:** the Loom architecture has been designed to facilitate the introduction of new types of knowledge structures and, in particular, to facilitate interaction with objects defined externally (e.g., LISP objects or objects defined in a relational database).

11.3 SCIENTIFIC PROGRESS

Early work on knowledge representation at ISI focused on the development of a system called NIKL [3]. In 1986, a workshop attended by users of NIKL was held to determine what needs were not being met by the NIKL system [7]. The requirements that emerged included (1) increasing the *flexibility* of the system to permit redefinition of terms already installed in a NIKL network; (2) extending the term-definition language to permit definitions of sets, sequences, and inverse and transitive relations; (3) the addition of a *constraint language,* which would permit term definitions to be augmented with specifications of necessary or sufficient conditions; (4) the addition to NIKL of a component for representing and reasoning with *factual knowledge.*

In mid-1986 the Empirically Valid Knowledge Representation project began developing an architecture to satisfy these new requirements. It was first determined that the existing NIKL system could not be simply upgraded. Accordingly, a new system was designed and built, which incorporated NIKL's major reasoning component, its *classifier*, but improved upon NIKL by modularizing the classifier and using more flexible data structures. This new system is called Loom [5].

The necessary flexibility was achieved in several ways: The Loom classifier is *incremental*, which means that whenever a user redefines a term in a Loom network, the system automatically reclassifies that term and all other terms that reference it. Second, Loom is implemented in CLOS, which makes it highly portable. The adoption of an object-oriented implementation makes it easy to define new types of objects and behaviors. This was demonstrated by adding to Loom the ability to define and classify inverse and transitive relations. Third, we implemented a subsystem that manages multiple knowledge bases organized in a hierarchy. Using this facility, knowledge bases can be shared (inherited) across multiple applications or they can be managed independently.

We developed and implemented a constraint language for Loom, which makes it the first classifier-based system able to handle a wider class of knowledge than just definitional knowledge. The constraint facility will support a style of programming called *constraint-based programming*: during the process of constructing a Loom model, the system will automatically detect any constraints implied by the partially constructed model. These constraints can be used to guide the selection of the next component in the model, and to inform the user when his/her model contains components that are inconsistent. We have a published a technical report [4] that contains an illustration of the constraint-based modeling process.

Work is underway in designing and implementing an assertional capability for Loom. We have implemented a database facility that stores factual information as CLOS objects, and a query facility that permits objects to be retrieved using a non-procedural query language. The next step will be to tie together the network and database components of the system. This will be accomplished by a pattern-matching facility that will dynamically compute the type of each database object by matching it against existing term definitions.

11.4 IMPACT

The development of NIKL led to a rapid increase in the number of research efforts using KL-ONE-like knowledge representation languages. The Consul user interface management effort was the first to use NIKL [6]. User interface research has continued with the use of NIKL in a multi-modal user interface system [1]. Natural language research has seen NIKL applied in a text-generation system [8] and another natural language understanding system [10]. One of the healthiest signs of NIKL's success is the number of knowledge representation researchers who have used NIKL as a starting point for their work. For example, KL-TWO uses NIKL as a T-Box to which it added an A-Box [9], and Ron Brachman has related that NIKL is the standard by which he has measured the Krypton T-Box [2].

Loom became operational in the last quarter of 1987, and ISI's Penman system was demonstrated using Loom in December 1987. In 1988, a new generation of applications is targeting Loom as the starting point for their work. The FAST-Workstation, BEAMER, and FIX projects at ISI are planning systems based on Loom. Additionally, ISI's Integrated Interfaces and SIMS projects are planning conversions to Loom later in 1988. The accomplishments already cited in Section 11.3, together with Loom's soon-to-be-implemented pattern-matching facility, will make available to these users a new dimension of capability that will significantly assist their programming and modeling efforts.

Numerous sites have requested copies of the NIKL system and, to date, NIKL has been exported to 17 sites outside of ISI. When Loom has reached an equivalent level of maturity, ISI will be releasing it to outside sites as well.

11.5 FUTURE WORK

ISI will continue the Knowledge Representation project under a follow-on contract. A major miletone of this effort will be the addition of a pattern-matcher and a truth-maintenance subsystem to Loom. Together, these new inference mechanisms accomplish the integration of terminological and constraint knowledge, stored in a Loom network, with factual knowledge stored in a database. The planned Loom pattern matcher will be more powerful than conventional pattern matchers because it will incorporate logical deduction into the match operation. This will enable users to express their patterns much more concisely than is possible in current-generation pattern-matching systems.

We plan to extend the representational capabilities of the system to provide specialized syntax and inference methods for sets, arithmetic intervals, and sequences. The system will also provide encapsulation mechanisms that will allow users to define Loom relations and functions that contain evaluation methods phrased in arbitrary LISP code. Methods will be developed to allow programmers to store externally defined LISP objects into a Loom knowledge base. The end result will be that Loom will support both high-level and low-level interaction with the complete spectrum of objects found in a LISP application environment.

A weak point in most knowledge representation systems, including NIKL and the present Loom system, is that they create too sharp a boundary between knowledge residing inside the system and knowledge residing outside the system. The mechani^ms just mentioned address a part of this problem. However, a complete solution to the problem will have to address the problem of how to represent *control* knowledge. Accordingly, Loom is embarking on a transition that will convert it from a knowledge representation system into a full-scale programming language. The focus of this work will be to rethink traditional programming paradigms such as those found in production rule and message-passing languages in light of the rich range of capabilities made available by Loom, and to evolve new programming styles wherein a program's control decisions fully exploit the knowledge representation subsystem's reasoning capabilities.

REFERENCES

1. Arens, Y., L. Miller, and N. Sondheimer, Presentation Planning Using an Integrated Knowledge Base, 1987. Submitted for publication.

2. Brachman, R. J., V. P. Gilbert, and H. J. Levesque, "An essential hybrid reasoning system: Knowledge and symbol level accounts of KRYPTON," in *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pp. 532-539, Los Angeles, August 1985.

3. Kaczmarek, T., R. Bates, and G. Robins, "Recent developments in NIKL," in *AAAI-86, Proceedings of the National Conference on Artificial Intelligence*, AAAI, Philadelphia, August 1986.

4. Mac Gregor, R., *The Knowledge Representation Project*, USC/Information Sciences Institute, RR-87-199, August 1987.

5. Mac Gregor, R., and R. Bates, *The Loom Knowledge Representation Language*, USC/Information Sciences Institute, RS-87-188, May 1987.

6. Mark, W., "Representation and inference in the Consul system," in *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, IJCAI, 1981.

7. Moore, J. D., NIKL Workshop Summary, 1986.

8. Sondheimer, N., and B. Nebel, "A logical-form and knowledge-base design for natural language generation," in *AAAI-86, Proceedings of the National Conference on Artificial Intelligence*, AAAI, Philadelphia, August 1986.

9. Vilain, M., "The restricted language architecture of a hybrid representation system," in *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pp. 547-551, Los Angeles, August 1985.

10. Weischedel, R., D. Ayuso, A. Haas, E. Hinrichs, R. Scha, V. Shaked, and D. Stallard, *Research and Development in Natural Language Understanding as Part of the Strategic Computing Program*, BBN Laboratories Incorporated, Cambridge, MA, Technical Report 6522, June 1987.

12. NATURAL LANGUAGE GENERATION

	***Research Staff*:**	***Support Staff*:**
William Mann	Robert Albano	Heidi Julian
Norman Sondheimer	Susanna Cumming	
John Bateman	Tom Galloway	
Eduard Hovy	Richard Whitney	
Robert Kasper		
Christian Matthiessen		
Lynn Poulton		

12.1 PROBLEM BEING SOLVED

The U.S. military is an information-rich, computer-intensive organization. It needs easy, understandable access to a wide variety of information. Currently, information is often expressed in obscure computer notations that cannot be understood without extensive training and practice. Although easy discourse between the user and the machine is an important objective for any situation, this issue is especially critical in regard to automated decision aids such as expert-system-based battle management systems that must carry on a dialog with a force commander. A commander cannot afford to miss important information, and it is unreasonable to expect force commanders to undergo highly specialized training in order to allow them to understand obscure computer dialects that change from machine to machine.

A great deal of work has been done in the area of natural language understanding, and this work is starting to pay off with the delivery of functional interfaces that interact naturally with the user. Comparatively little, however, has been done in the area of natural language *generation*. There is currently no effective technology for expressing complex computer notations in ordinary English. If there were, computer-based information could be made more accessible and understandable in a way that was less subject to personnel changes.

12.2 GOALS AND APPROACH

Penman is a natural language sentence generation program being developed at ISI. It provides computational technology for generating English sentences and paragraphs, starting with input specifications of a non-linguistic kind.

The research objectives underlying Penman are threefold: to provide a useful and theoretically motivated computational resource for other research and development groups and the computational community at large, to provide a framework in which to conduct investigations into the nature of language, and to provide a text generation system that can be used routinely by system developers. Penman is being used by computer scientists (as the output medium of their programs, including projects in human-computer communication, expert system explanation, and interface design, among others) and by linguists (as a reference and research tool).

This enterprise can be divided into two parts: the development of single-sentence generation technology and the development of multisentence planning technology. The Penman project has made a significant achievement with the first part and has recently started making great progress with the second.

12.3 SCIENTIFIC PROGRESS

This project was put into place as part of DARPA's Strategic Computing Program, intended to be used in the battle management system developed under the Fleet Command Center Battle Management Program (FCCBMP). This report covers the last half of 1986 and the year 1987. During this time, we made the following progress:

- **Joint work with BBN:** This period saw the culmination of the joint effort with Bolt Beranek and Newman, Inc. (BBN) of Boston to parse natural language queries to a Navy database and to generate the answers with Penman. The combined parser and generator, called Janus, used common representations and a common domain model. A demonstration of this capability was held in Philadelphia in May 1987.
- **Multisentential text planning:** During 1987, we started work on the planning of multisentence texts by computer. This work involved the formalization of the interclausal relations of Rhetorical Structure Theory and their operationalization as plans in a standard hierarchical expansion planner. It also required the interfacing of the planner's output with Penman, using the verbalization graph notation. Both the planner and the formalization work are still undergoing development and refinement, but we have already generated paragraphs of text for two distinct domains. The planning work is described below. A paper about this work has been submitted to the 1988 meeting of the ACL conference.
- **Integrated Interfaces (II) support:** To find the first testbed for the new multisentence text planner, we collaborated with the Integrated Interfaces project. Within a period of some months we were able to generate a number of paragraphs containing information derived from the Navy database. These paragraphs were incorporated into the II display as blocks of text superposed on a map, tied to ships or ports. In addition, we held a number of meetings with Navy personnel in order to produce exactly the kind of language they used. This experiment was very successful, judging by the reactions of the Navy personnel who saw the II system in action, as well as the reactions from the interfaces community.
- **Explainable Expert Systems (EES) support:** After the initiation of generating multisentential text for the Integrated Interfaces application, we started work on generating text for the expert system PEA (Program Enhancement Advisor), part of the Explainable Expert Systems project at

ISI. The goal of this expert system is to describe its results and explain its reasoning in natural language, in an interactive manner, allowing the user to query it on points not understood. Since the text planner for PEA was not yet developed, we offered to provide not only the single-sentence generation capability, but also the multisentential planning capability. This offer was accepted and led to much fruitful collaboration, as described below.

- **Grammar extension:** A number of extensions to the Nigel grammar were completed during 1987. Some of these extensions were prompted by the new demands being made on Penman as a result of generating multisentential text: notably, the ability to generate interclausal linking expressions such as "in order to."
- **Lexical choice work:** Some project members collaborated on developing an algorithm for more sophisticated lexical choice than had been implemented before. A description of this work has been accepted for presentation at the lexicon workshop to be held early next year in Boston.
- **Nigel distribution:** The Nigel grammar was distributed to three sites -- Columbia University, the Machine Translation Center at Carnegie-Mellon University, and (in paper form, due to licensing problems with IBM) to the IBM natural language research center in Los Angeles. We are also actively searching for other suitable sites to which to send Nigel, both in the U.S. and abroad, particularly in Toronto, where linguists at York University have expressed some interest in using it as a reference tool.
- **Nigel documentation:** We continued writing the Nigel documentation, an exhaustive description of the more than 500 grammatical systems in Nigel, together with their associated choice functions, defaults, etc.

Two principal developments occurred in the Penman project during this period. The first of these was the culmination of the collaboration with BBN on Janus, the English-in English-out database question-answering system, and the second was the development and use of a prototype multisentential text planner. Additional regions of continuing interest and research were the extension of the capabilities of the Nigel grammar and the streamlining and speeding up of the Penman program.

In May 1987 Janus was demonstrated at a DARPA meeting in Philadelphia. The Janus parser was developed by BBN; the language generation was handled by Penman. The internal representation language and domain model (the model contained information about Navy ships, statuses, and actions) were shared by the two programs. Several types of queries were parsed and a number of types of responses were generated, demonstrating the feasibility of English-in-English-out technology. At this point the collaboration had run its course, and both partners moved into different areas of research.

A major portion of the rest of the year was spent in developing multisentential text-planning technology and in getting that technology linked to Penman and to two disparate application domains. As a result, Penman generated paragraphs of text for a multimodal Navy database information display system and a self-explaining expert system.

Development of the text-planning technology was engendered by the wide-ranging program of research analyzing the structure of coherent paragraphs which resulted in Rhetorical Structure Theory (RST). Dr. Bill Mann of the Penman project and Prof. Sandra Thompson of UC Santa Barbara had analyzed hundreds of paragraphs and proposed that a set of 20 relations suffice to represent the relationships that hold within texts that normally occur in English. These relations are interpreted recursively, the assumption being that a paragraph is only coherent if all its parts can eventually be made to fit under one overarching relation.

Under a slightly more operational and goal-oriented interpretation, these relations can be seen as plans that govern the assembly of clauses into coherent paragraphs. Dr. Eduard Hovy developed a top-down hierarchical planner, similar to the well-known planner NOAH, which employed these relation/plans to structure paragraphs from given collections of input.

This method of planning paragraphs afforded much more flexibility of assembly than previous methods allowed. In this sense, the development of multisentential planning technology paralleled the development of single-sentence generation technology. The earliest techniques for producing single sentences by computer relied on the canned text and template methods, in which either a predefined string stating the information was selected and output, or slots of a predefined template were filled in with appropriate aspects. Though useful in very limited domains, the lack of an intelligently controlled construction process imposed severe limitations on the flexibility and extensibility of such techniques. In the last decade, work on sentence generation has produced progressively more refined generators, to the point where the most powerful generators today dynamically assemble large numbers of very detailed grammatical features that together specify a sentence.

The work on the production of multisentential paragraphs has had a similar history, with a roughly 10-year lag time. The first systems to produce paragraphs of text used so-called schemas that described the content and order of the clauses of the paragraph. Each schema was a static representation of a particular discourse strategy that people typically employ in conversation; each schema thus produced a different type of paragraph. Though early schemas afforded some variation, and later schemas were built to accommodate additional types of variation, these structures in general suffer from the same lack of flexibility that hampered sentence templates.

During the last year, the Penman project has been formalizing RST relations and using them generatively to plan paragraphs. Relations are seen as plans -- the operators that guide the search through the permutation space of the input units. Constraints on the parts of the relation/plans become requirements that must be met by any piece of input before it can be used in the relation (i.e., before it can be coherently juxtaposed with the preceding text). The effects of relation/plans are descriptions of the intended effect of the relation (i.e., the communicative goal that the relation achieves, if properly executed). Since the goals in generation are communicative, the intended effect must be seen as the inferences that the speaker is licensed to make about the hearer's knowledge after the successful completion of the relation/plan. The constraints and effects of plans are represented in terms of the formal theory of rational interaction currently being developed by (among others) Cohen, Levesque, and Perrault.

The text-structure planner operates antecedent to Penman. It plans coherent paragraphs to achieve communicative goals posted by the user's system to affect the hearer's knowledge in some way. It accepts one or more inputs from the domain of discourse and rewrites the inputs into a common form (called here input units) that consist of collections of input characteristics. Then, by planning, it assembles the input units into a tree that expresses the paragraph structure. Finally, the planner traverses the tree, dispatching the leaves (the input unit clauses) to be generated by Penman. During traversal of the tree, the planner performs additional planning tasks, such as sentence size delimitation and focus control.

The planner embodies a limited *top-down hierarchical expansion planning framework*. Each relation/plan has two parts, a *nucleus* and a *satellite*, and relates some unit(s) of the input or another relation (cast as nucleus) to other unit(s) of the input or another relation (cast as satellite) recursively. In order to admit only properly formed relations, nuclei and satellites contain requirements that must be matched by characteristics of the input. Thus, for example, the PURPOSE relation/plan cannot be used to relate some input state or condition to some input action unless it can be proved (to the planner's satisfaction, using the PURPOSE requirements) that the state was in fact the purpose of the action.

A few papers relevant to the use of RST and the new planning technology are listed below.

- Cohen, P. R., and H. J. Levesque, "Speech acts and rationality," in *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, Austin, Texas, 1985.
- Hovy, E.H., "Planning Coherent Multisentential Text," submitted to the Meeting of the Association for Computational Linguistics, to be held in 1988.
- Mann, W. C., and S. A. Thompson, "Assertions from discourse structure," in *Proceedings of the Eleventh Annual Meeting of the Berkeley Linguistics Society*, 1985. Also available as USC/Information Sciences Institute, RS-85-155, April 1985.

- Mann, W. C., and S. A. Thompson, "Rhetorical Structure Theory: Description and construction of text structures," in G. Kempen (ed.), *Natural Language Generation: New Results in Artificial Intelligence, Psychology, and Linguistics*, Martinus Nijhoff Press, 1987.
- Sacerdoti, E., *A Structure for Plans and Behavior,* North-Holland, 1977.

The multisentential planning technology was tested on two disparate application domains: Integrated Interfaces (II) and the Program Enhancement Advisor (PEA).

The Integrated Interfaces project at ISI is developing a prototype interface management system that handles mixed-mode input (menus, forms, pointing) and incorporates a combination of output modes (NL text, maps, menus, and forms). Integrated Interfaces uses Artificial Intelligence knowledge base and rule technology to link together knowledge of an application domain with facilities of the user interface. A frame-based knowledge representation system (LOOM or NIKL) is used to model the entities of the application domain and the facilities of the user interface. These domain and interface models are related by antecedent-consequent rules to determine appropriate methods of displaying information.

The II project has implemented a demonstration interface to an existing Naval database reporting application. This interface system creates displays similar to those being prepared manually for the Navy on a daily basis. In these displays, Penman generated natural language texts, which were placed at appropriate locations on a map to describe the activities of important objects, such as ships.

Within Integrated Interfaces, our aim was to provide a component that would present, in natural language, the information that was suited to language and not suited to two-dimensional display modes such as maps or tables. Our task was to accept from the II display manager the information to be generated, to structure this information into coherent paragraphs, and to generate the English sentences comprising the paragraph. The information is provided in the same representation scheme used by the rest of the II system; no special language-based alterations are made. Thus the Penman text-structure planner and sentence generator act as a subsystem of II. Our task consisted of three principal parts:

1. the prestructuring of the input into individual clause-sized units;
2. the construction of a coherent paragraph, including appropriate interclausal relation words; and
3. the generation of individual clauses in English.

Task 1 required domain-specific information. We built a module that used Navy rules to group the information provided by II into clause-sized units and to extract from the information such events as arrivals, departures, and rendezvous events. (This information is not explicitly represented in II. In order to be generated, however, it must be given explicit status. For example, arrival events were created using the rule [*if a mobile employment is followed by a stationary employment, then an ARRIVE event occurs between them*]).

For Task 2, the text-structure planner was used to build a coherent paragraph from the clause-sized units. As described, RST relation/plans provided the constraints necessary to impose coherence and also provided typical interclausal conjunctive words and phrases where useful.

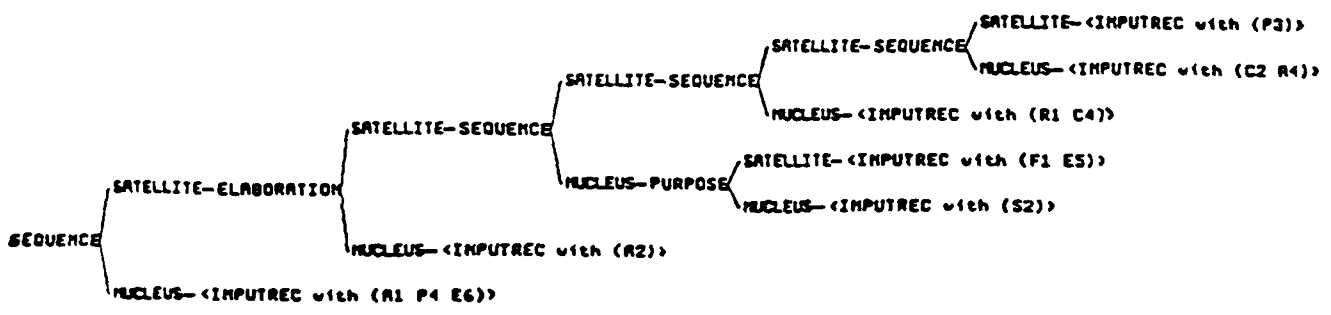
For Task 3, a tree-traversal algorithm gathered the clause-sized chunks of input contained in each leaf of the tree and activated Penman with these chunks to produce a list of sentences. The paragraph of sentences was then returned to the Integrated Interfaces display manager to be formatted and presented on the screen inside a text block. The following paragraphs are a few of those generated for this domain:

```
Knox, which is C4, is at 79N 18E heading SSW. It is en
route to Sasebo, arriving 4/24, in order to load for four
days.

Knox is en route to Sasebo. It will arrive 10/24. It will
load until 10/28.

Knox and Fanning are en route to Sasebo, arriving 4/24.
While it is in Sasebo, Knox, which is C4, will load until
4/26. Fanning will depart on 4/25 in order to rendezvous
with CTG 73.1 on 4/28.

Fanning, Passumpsic and Whipple are en route to rendezvous
with CTG 070.10, arriving tomorrow. Fanning and Whipple will
be on operations until 10/26. Passumpsic will be performing
services until 10/28.

New Jersey, Copeland and Merrill are on operations until
4/20. Thach is on operations until 4/21.

MEKAR-87 takes place with Knox, Fanning, and Whipple in
South China Sea from 10/20 to 11/13. Knox and Fanning
join 10/20. Whipple, which is C4, joins 10/29. Knox
departs on 10/31. Fanning and Whipple depart on 11/13.
```

The result has been very successful. The Integrated Interfaces system has been demonstrated to a number of Navy officials, among others the personnel responsible for creating the Navy's daily briefings, which consist of maps showing ship positions and descriptions of their employments. They currently perform this task by hand. Their reaction was extremely positive, to the point where funding is currently being negotiated to produce a full-scale system for use by the Navy.

From our perspective, we found the Integrated Interfaces experiment very useful in identifying the difficult problems inherent in paragraph planning, and in finding the points where further theoretical work is indicated. For example, we believe that the planning of page-length reports of the activities of ships and of the visitors to ports is now a feasible undertaking. This would be the first time ever that a computer has planned and generated more than two or three relatively short paragraphs of connected text.

The second testbed for the multisentential planning technology was an expert system, part of the Explainable Expert Systems project. The Program Enhancement Advisor expert system (PEA) suggests improvements to LISP programs and interactively explains its recommendations and its reasoning to the user. We are collaborating with PEA in order to produce text that is satisfactory for its needs. We have not yet completed the modeling of the domain knowledge in terms Penman understands; however, to date we have produced a handful of paragraphs, of which the following text and associated paragraph structure tree is an example:


SEQUENCE
SATELLITE–ELABORATION
SATELLITE–SEQUENCE
SATELLITE–SEQUENCE
SATELLITE–SEQUENCE
SATELLITE–<INPUTREC with (P3)>
NUCLEUS–<INPUTREC with (C2 R4)>
NUCLEUS–<INPUTREC with (R1 C4)>
NUCLEUS–PURPOSE
SATELLITE–<INPUTREC with (F1 E5)>
NUCLEUS–<INPUTREC with (S2)>
NUCLEUS–<INPUTREC with (R2)>
NUCLEUS–<INPUTREC with (R1 P4 E6)>

```
The system asks the user to tell it the characteristic of the
program to be enhanced. Then the system applies transformations
to the program. In particular, the system scans the program in
order to find opportunities to apply transformations to the
program. Then the system resolves conflicts. It confirms the
enhancement with the user. Finally it performs the enhancement.
```

12.4 IMPACT

During the reporting period, project members attended the following conferences and workshops, and presented the following papers:

American Association of Artificial Intelligence (AAAI, Philadelphia, PA, August 1986):

- Norman Sondheimer and Bernhard Nebel, "A Logical-Form and Knowledge-Base Design for Natural Language Generation."

Third International Workshop on Language Generation (Nijmegen, The Netherlands, August 1986):

- William Mann and Sandra Thompson, "Rhetorical Structure Theory: Description and Construction of Text Structures."
- Christian Matthiessen, "Notes on the Organization of the Environment of a Text Generation Grammar."

International Workshop on the Lexicon (Pisa, Italy, August 1986):

- Susanna Cumming, "The Lexicon in Text Generation."

Workshop on Future Directions in Artificial Intelligence (U.S. Army Research Office, October 1986):

- William Mann, "Text Generation: Is It the Graphics of the 90s?"

Second Annual Meeting of the Pacific Linguistics Conference (November 1986):

- Sandra Thompson and William Mann, "A Discourse View of Concession in Written English."

Applied Conference of the Association for Computational Linguistics (ACL, Austin, TX, February 1987): Dr. Norman Sondheimer was the chair of the organizing committee of the conference.

TINLAP-3 Workshop (Las Cruces, New Mexico, February 1987): Five project members attended the workshop. Drs. William Mann and Norman Sondheimer were members of panels.

DARPA Workshop (Philadelphia, May 1987): Four project members attended the DARPA workshop, during which the joint BBN-ISI natural language dialogue system Janus was demonstrated.

Association for Computational Linguistics (ACL, Palo Alto, June 1987): Four project members attended the ACL conference. Dr. Mann, the then president of the Association, gave the banquet address.

- Robert Kasper, "A Unification Method for Disjunctive Feature Descriptions."

Fourteenth International Systemics Workshop (Sydney, Australia, August 1987): Two project members, Dr. Bill Mann and Mr. Christian Matthiessen, attended the workshop, where each delivered a paper. Dr. Bateman also attended the workshop, although he had not yet joined the project.

- William Mann, "Toward a Theory of Reading Between the Lines."
- Christian Matthiessen and William Mann, "Functions of Language in Two Frameworks."

American Association of Artificial Intelligence (AAAI, Seattle, August 1987): Three members of the project attended the conference.

- Eduard Hovy, "Interpretation in Generation."

Ninth Annual Meeting of the Cognitive Science Society (Seattle, August 1987):

- Eduard Hovy, "What Makes Language Formal?"

DARPA Workshop (Palo Alto, November 1987): Three project members attended the DARPA evaluation workshop and presented two papers.

The multisentence text-planning technology we developed has already been picked up and taken a step further by two NLP researchers in the Explainable Expert Systems project at ISI. Dr. Cecile Paris and Ms. Johanna Moore have built a second, more powerful version of the planner, which uses an enlarged set of relation/plans to control the search for information to be displayed before structuring it into coherent paragraphs. This is an exciting development, directly addressing one of the possible avenues of research opened up by the use of RST in text generation. Their research is now complementary and parallel to ours, in the sense that they are investigating the use of relatively fixed relation/plans, similar in essence to schemas, and we are continuing our investigation of a more flexible planning regime that involves the use of dynamic criteria for including and ordering material. We anticipate that this parallel research will be beneficial to both sides and will serve to investigate this aspect of text planning thoroughly and speedily.

12.5 FUTURE WORK

This research will continue a a follow-on project. Our planned activities for the next year include the following:

- **Penman distribution.** It is our intent to distribute Penman--that is, Nigel together with its auxiliary information sources such as the Upper Model--in the coming year. In order to perform this distribution, we need to write documentation of Penman to accompany the Nigel Manual.
- **Penman input notation.** Our experience with II and PEA has led us to believe that the current input notations used by Penman are unnecessarily complex and particularly hard to read. We will investigate this matter with the possibility of designing a new input notation.
- **Multisentential text planning.** We will continue to enhance the planning capabilities of the paragraph planner, by formalizing and implementing additional Rhetorical Structure Theory planning operators and by incorporating solutions to new problems that arise.
- **Integrated Interfaces support.** We will continue providing support for the II project, aiming toward full coverage of the language required by the contents of the database.
- **Explainable Expert Systems (EES) support.** We will continue our support of PEA, aiming toward full coverage of its needs. We will generate text for any new EES systems that may be developed.

Since Drs. Bateman and Hovy have joined ISI, the Penman project, together with members of the EES project (Dr. Bill Swartout and Ms. Johanna Moore), constitutes the largest grouping at a single institution of trained Natural Language Generation researchers in the world. We expect that the coming years will be exciting times for the project, times in which great progress will be made.

The development of the multisentence text planner and its successful linking to two different domains has opened up new horizons for Penman. It has also indicated a whole new set of problems that require further work. We expect that Penman will continue to grow in scope as such additional functionality is added, and we believe that the core of the system, the Nigel grammar, represents a sound foundation on which to base further growth.

SELECTED PUBLICTIONS

Book Chapters

1. Mann, William, and Sandra Thompson, "Rhetorical Structure Theory: Description and construction of text structures, in G. Kempen (ed), *Natural Language Generation: Recent Advances in Artificial Intelligence, Psychology, and Linguistics*, Kluwer Academic Publishers, 1987.
2. Thompson, Sandra, and William Mann, "Antithesis: A study in clause combining and discourse structure," in R. Steele and T. Threadgold (eds.), *Language Topics: Essays in Honour of M.A.K. Halliday*, Benjamins, 1987.

Journal Articles

1. Mann, William, and Sandra Thompson, "Relational propositions in discourse," in *Discourse Processes,* 9 (1), pp. 57-90, 1986.
2. Hovy, Eduard, "Generating natural language under pragmatic constraints," in *Journal of Pragmatics*, XI (6), pp. 689-719, 1987.
3. Mann, William, and Sandra Thompson, :Rhetorical Structure Theory: A framework for the analysis of texts,: in *IPRA Papers in Pragmatics* 1, pp. 79-105, 1987.

13. COMMERCIAL MAIL

Research Staff:	*Support Staff*:
Dale Chase	Manon Levenberg
Craig Ward	Glen Gauthier

13.1 PROBLEM BEING SOLVED

The evolution of large electronic mail systems testifies to the increasing importance of electronic mail as a means of communication and coordination throughout the scientific research community. These systems include the DARPA Internet mail system, the GTE Telemail system, the MCI-Mail system, and the IEEE Compmail system on ITT Dialcom. Until now these systems have operated autonomously, and no convenient mechanism has existed to allow users of one system to send electronic mail to users on another system. The Intermail system, developed by the Internet Concepts Research project at ISI, demonstrated a mail-forwarding system that allows users to send electronic mail across such mail system boundaries. The Commercial Mail project will convert this system from a research project into a commercial product.

13.2 GOALS AND APPROACH

The most significant limitation of the Intermail system is its inability to handle forwarding to more than one "other" mail system in a single interaction--that is, each message can be delivered to only one other mail system. We will eliminate this limitation by introducing a new syntax for addresses. Any number of addressees may be specified on any number of "other" mail systems. This will allow users on "other" mail systems to be included in mailing lists along with DARPA Internet recipients, UUCP recipients, CSNET recipients, etc.

It is presently possible for DARPA Internet users to communicate easily with users of both UUCP and CSNET via cooperating hosts that maintain connections with these communities. Addresses are specified in the local syntax of each of the three systems, and conversions are handled by the forwarding hosts. This is the way we intend to have the commercial mail forwarding facility function. This is a relatively easy facility to provide on the DARPA side, where the address syntax is well understood and flexible, and the entire process will be under our control. On the commercial side, we will continue to use the techniques developed by the Internet Concepts project, which use forwarding information embedded in the text portion of the message in a Simple Forwarding Header (SFH). Our conversion system will, however, make it possible for DARPA Mail recipients to reply directly to messages from commercial users using existing DARPA Mail composition programs.

13.3 SCIENTIFIC PROGRESS

Upon the successful development of the Intermail system, the ISI computer center software group was contracted to build a robust, production-quality, operating Commercial Mail Relay (CMR) system. In its first year the CMR project spent a great amount of time evaluating the Intermail research project software, various operating system platforms, and network mail system software suitable for building the foundation of a commercial product. With the selection of 4.3 bsd UNIX for the operating system and the Multi-channel Memo Distribution Facility (MMDF) as the system mailer, the first specifications for the CMR were written.

UNIX and MMDF were selected for system software so that the CMR could be developed in a highly portable environment. Both UNIX and MMDF run on a variety of platforms. MMDF, with its notion of a dedicated channel for messages going to a particular destination, allows great flexibility concerning the addition or removal of commercial networks.

Development began on a VAX/11-750 with the target production machine being a Microvax VAXStation II. During the development period, numerous demonstrations of the system were given, enabling researchers at ISI to give the CMR development team positive feedback. The first CMR MMDF channel and a pre-processor for SFHs were put into service in October 1987.

This channel bridges the GTE Telemail system to the Internet. It interfaces to the Telemail system by simulating a human using a terminal. Negotiations are under way with GTE to update their software to allow direct computer access. Because the GTE system assumes that a human is entering the system, mail errors (such as misaddressed mail) are handled in a verbose format. This makes error handling in CMR very challenging. The Telemail channel and the SFH pre-processor currently handle some failed mail situations, such as incorrect commercial mail system, incorrect commercial mail user name, and incorrect commercial mail host. We plan to include others and enhance the understanding of CMR-generated error messages as our experience with the CMR user community increases.

After the Telemail channel and SFH pre-processor stabilized, the CMR development team began designing a channel for ITT Dialcom. Several organizations, government and private, operate on Dialcom systems. The CMR Dialcom channel will be used by IEEE, ONR, and NSF.

While the design and coding of the Dialcom channel proceeded, bugs and other required changes were made to the Telemail channel and the SFH pre-processor, along with bug-fixes and upgrades for the system software. User documentation was updated and the program logic manual was edited to reflect changes and in response to user needs. Simple accounting programs were developed to track the usage.

Work continues on refining and expanding the Commercial Mail Relay.

13.4 IMPACT

The availability of high-performance personal workstations and dedicated special-purpose processors, and the preference for these machines by researchers, have made the acquisition and operation of large mainframes capable of providing reliable mail service unacceptably high. Since most of these research machines cannot provide mail service, the Commercial Mail implementation will allow the users of these machines to remain accessible to the rest of the research community without maintaining two distinct computing facilities.

The availability of a reliable high-capacity bridge between commercial mail systems and the DARPA Internet will provide better communication among contractors, especially those involved in the Strategic Computing Program, who might otherwise be denied convenient access to colleagues.

13.5 FUTURE WORK

ISI will continue to update and support the CMR system. We plan to add software to the system that will allow mail to be relayed from the Internet to Dialcom (for ONRMail, IEEE Compmail, and NSFMail) and MCI Mail. The CMR is flexible enough that the addition of other commercial mail networks such as Compuserve, GENIE, or AT&T Mail will b^ possible depending on the interest expressed by the Internet community in any such networks. ISI will also provide the following enhanced services:

- negotiate more cost-effective billing procedures from commercial networks
- negotiate the implementation of new machine interfaces with commercial mail networks
- improve systems reliability
- implement software that will allow faster file transfer
- improve systems accounting

The improved systems accounting software will include the ability to recharge costs to remote sites that use the CMR system.

Once this software is developed, ISI will serve as a distribution center for outside agencies that want to use this type of mail relay software. This software will be fully supported by a set of user documentation. ISI will maintain an electronic mailbox that can be used by remote users to report bugs, problems, etc. ISI will also distribute the CMR software to other sites that run UNIX.

14. COMPUTER RESEARCH SUPPORT

Director: Ronald Ohlander

Technical Staff:

Software:	***Hardware:***	***Operations and Network Services:***	
Dale Chase	Daniel Pederson	Stephen Dougherty	
David Bilkis	Val Fucich	Walt Edmison	Vicki Gordon
Dwight Fromm	Glen Gauthier	James Hurd	Kevin Haley
Bryan Howard	Ramon Gonzalez	Roylene Jones	Anna-Lena Neches
Jim Koda	Fred Grolle	Joe Kemp	Chris Refuerzo
William Moore	Raymond Mason	Roger Lewis	Wayne Tanner
Koji Okazaki	Daniel Trentham	Sean Schur	Robert Wormuth
Rod Van Meter		Toby Stanley	
Craig Ward		Christine Tomovich	
Tom Wisniewski		Mike Zonfrillo	

Support Staff:
- Manon Levenberg
  - Nancy Garman
  - Pat Thompson

14.1 BACKGROUND

ISI provides cost-effective and key computer support to researchers at ISI and DARPA, DARPA contractors and affiliates, and the military services. In addition to providing raw computing service on TOPS-20 to members of the DARPA research community and DARPA itself, ISI also originally developed and provided and/or now helps to support many additional mature software packages for the entire Internet community, including text editors (XED, Emacs), mail handlers (Hermes, MSG, SNDMSG, MM), compilers (PASCAL, C, FORTRAN, COBOL, MACRO, BLISS), language environments (Interlisp, Mainsail, Ada), and network access tools (Telnet, FTP, SMTP).

ISI has built a reputation for excellence and efficiency in both experimental computer services and production services, and has repeatedly demonstrated its continuing high standards to users while maintaining a relatively small expert staff. ISI provides guidance to DARPA on what constitutes a sensible load for a TOPS-20 host, and also provides management control to ensure an adequate level of support for the DARPA user community.

ISI has also shown itself to be extremely proficient in the realms of network access and host security, successfully walking the fine line between protecting the interests of the community and having that security be an encumbrance to the user community. Security involves a number of access controls and, perhaps more important, it includes protection schemas, password encryption checkers, and monitoring demons that can be activated when a problem is suspected.

ISI and its current research projects have benefited substantially from in-house competence in computer services, particularly as the original support of the TOPS-20/DEC-KL environment has widened to include an eclectic and complex collection of hardware and software architectures. The staff of the Computer Center has grown in expertise and maturity to meet the demands of this wider collection of equipment and software, while the total head count of the group has been reduced.

With the continuing evolution of computer workstation technology, ISI expects to see a decline in the demand for network-based, interactive, timesharing cycles as users move into workstation/local area network environments.

14.2 PROBLEMS BEING SOLVED

The Computer Research Support project is responsible for providing reliable computing facilities on a 24-hour, 7-day schedule to the Internet research and development community. At the same time, the project makes available to Internet users the most current releases of hardware and software on the supported machines. The project provides continuous computer center supervision and operation, and a full-time customer-service staff that responds to user inquiries. This project supports a major computer installation at ISI's main facility in Marina del Rey, California, but shared staff within the facility lends infrastructure support to ISI's internal research efforts.

14.3 GOALS AND APPROACHES

The ISI Information Processing Center provides support in four distinct, though tightly interrelated, areas: Hardware, Systems Software, Operations, and Network Services. The overall charter of the organization is to assure that the needs of the user community are addressed and protected as efficaciously as possible. To achieve this end, each group is concerned about the effective use of the machines and software tools, and about the security of the physical plant, the system files, and other online information. The more specific goals and approaches of each group are summarized below.

Hardware

To achieve a reliability goal of 98.7 percent scheduled uptime, preventive and remedial maintenance responsibilities have been assigned to an in-house computer maintenance group. This group provides cost-effective 20-hour, 5-day on-site coverage and on-call standby coverage for after hours. To maintain the reliability goals, preventive maintenance is very closely controlled, and online diagnostics and analysis are emphasized. A primary component in the reliability and availability of the hardware is the physical environment in the computer facility itself. Accordingly, significant time and resources are expended in ensuring that the best, most cost-effective environmental controls are at the facility.

System Software

The software group's overall goal is to install and maintain, at maximum reliability, ISI's VMS, UNIX, and TOPS-20 operating systems and applications software. In order to accomplish this goal, the group provides 24-hour, 7-day coverage to analyze system crashes and to provide appropriate fixes. In addition, it is the group's responsibility to install, debug, and modify the latest monitor and kernel versions, and the associated subsystems, available from the vendors.

Operations

The Operations staff is responsible for maintaining the computers on a day-to-day basis, and also for monitoring the systems and the physical environment of the computer room itself. Operations at the facility runs on a 24-hour, 7-day on-site schedule. One of the primary responsibilities of the group is to provide protection and backup for user files in order to ensure the integrity of all data. This goal is achieved through a variety of means, including regularly scheduled full and incremental backups of all systems; permanent archivals of requested or infrequently accessed system and user files; magnetic tape storage and pointers to all information extant at the time of removal of directories from the various systems; and, perhaps most important, redundant offsite storage of all significant information active on the disk structures or existing on tape within the facility.

When a problem occurs, the on-duty staff isolates it and takes appropriate action. They work closely with the on-call hardware and software support staff in resolving system problems so that maximum uptime can be achieved. On the night and weekend shifts, the Operations staff responds directly to user inquiries. Proper training, experience, continuity, and familiarity with the environment are especially stressed.

Network Services

Network Services, the ISI customer-service group, provides a two-way communication link between the users and the rest of the support staff. The group handles all directory management issues, monitors available disk space, answers questions about the primary software packages, and solves a number of lower level technical questions. This support is accomplished by maintaining a 12-hour, 5-day on-duty staff for prompt problem resolution and rapid information exchange, both on-line and by telephone. The group also offers introductory training in the use of hardware and software tools available on the ISI systems, as well as providing documentation for new users of the Internet. Network Services also assists in the formulation of user training programs for large, Internet-based military experiments at, for example, the Strategic Air Command, Offutt Air Force Base, Nebraska; the Naval Postgraduate School, Monterey, California; and the Systems Design Center at Gunter Air Force Base, Alabama.

Appropriate documentation is constantly being generated and distributed to individual users, as well as to remote user-group liaison personnel; this documentation ranges from simple, beginner-level explanations to more technical information suitable for experienced users. In accordance with ISTO guidelines, the customer-service group provides regular system utilization accounting data to DARPA.

14.4 PROGRESS

The ISI Computer Center Facility operated without major hardware or software difficulties during the reporting period. Scheduled uptimes averaged well above 99 percent. One of the six DEC PDP-10 computers running the TOPS-20 operating system was shut down in October 1986. This system was used in support of the ARPANET TAC Access project. These functions were transitioned to BBN with virtual transparency. The development of plans to move users and then shut down three additional DEC PDP-10 hosts also began during this period. The users of these systems will be transferred onto VAXes and a variety of workstations.

Hardware Additions

The ISI Computer Center continued its move away from DEC PDP-10 computers toward personal workstations and file servers. The facility acquired a large number of workstations and a DEC VAX 8650 running BSD 4.X UNIX.

The hardware staff performed upgrades and additions to the hardware configurations on several of the systems (including security improvements). An additional change included more memory on ISIB. Most of the old RP06 disk drives were phased out in favor of larger, more modern, and cost-effective Winchester technology RP07 drives on all systems. The RP06 drives that remain are now mostly used for mountable structures. All installations and changes were carried out during scheduled evening maintenance hours without interfering with normal operations.

The Computer Center houses and/or lends substantial support to the array of personal computers (PCs), workstations, and symbolic processing engines acquired over the last several years by a variety of projects. The Center assures that the filesystems of these various machines are backed up and that network connectivity (where required) is robust.

The current list of major multiuser processors, network servers, and supported individual machines (PCs, workstations, and symbolic processing computers) follows:

10 DEC VAX 11/750 computers
3 DEC VAX 11/780 computers
2 DEC VAX 8650 computers
10 DEC MicroVax computers
19 Xerox Dandelion processors
3 Xerox 1100 LISP machines
18 Symbolics 3600/3645/3675 LISP machines
35 IBM-PCs (some XTs & ATs)
38 SUN Microsystems 68010/68020 systems
20 Hewlett-Packard Bobcat workstations
17 Texas Instruments Explorer workstations
10 Apple IIE/Lisa/Macintosh computers
2 Iris workstations
120 modems (300, 1200, 2400 Baud)
220 computer terminals (HP, DEC, Zenith, etc.)

1 Micom data switch (2 bays)
1 LeeMah data security switch
1 Xerox 5400L Penguin laser printer
1 Dataproducts high-speed printer
2 Xerox 2700 laser printers
4 Imagen laser printers
1 Xerox 8045 print server
2 Xerox 8031 file servers
6 HP Inkjet desktop printers
22 Epson desktop printers
3 BBN Butterfly gateways
2 Grid computers
3 Three Rivers PERQ computers

System Software Enhancements

Testing of the Stanford/CISCO Massbus Ethernet Interconnect System (MEIS) was completed. The remaining TOPS-20 systems were upgraded to the 6.1 Monitor, which included the incorporation of the MEIS software. Domain naming of all ISI hosts was also accomplished.

14.5 MILITARY IMPACT

ISI is perhaps the finest university-based research center promoting the sharing of software resources within the DARPA community; it assumes responsibility for providing support to key DoD community personnel so as to demonstrate the great utility of the ARPANET and MILNET resources. The effective and rapid transfer of information, electronic mail and other data, computer programs and tools illustrates, on a working daily basis, ways to enhance military efficiency. Specific technology transfer of relevant research to the military is heavily dependent on the facility.

ISI's Computer Center provides ARPANET/MILNET cycles and support 24 hours a day, 7 days a week to the Strategic Air Command, Naval Postgraduate School, Gunter Air Force Base, the Office of Naval Research, and the ADDS Experimental Test

Division at Fort Bragg, North Carolina, as well as to the contractors, researchers, and administrators of militarily significant research coordinated out of the DARPA office in Washington, D.C. In addition to supplying machine time and directed Network Services support, this project continues to provide substantial additional support in the following areas:

- General accounting information as needed for workload analysis on the various machines.
- Rapid response to user requests and problem reports.
- Tailored security awareness, and manual and automated tracking.
- Maintenance and upgrading of electronic mail system software, shared online bulletin boards, and other specialized communications and file-transfer programs.
- Maintenance of approximately 2500 user directories (as well as hundreds of support and overhead directories) on the TOPS-20 machines used by the DoD and affiliates.

14.6 FUTURE WORK

The Computer Research Support project will continue to provide computing service to the DARPA research community, provide and support software packages for the Internet community, and offer a program of technology transfer and user education through the Network Services group.

Some specific planned activities for the next year include the following:

- The move of local ISI staff to the NCE (New Computing Environment), resulting in the potential freeing up of the last TOPS-20 resources to the wider military community.
- Installation of new procedures and support for acquisitions of the NCE and other projects at ISI. This will include potential new products (either purchased at DARPA's direction or via grants) from the following vendors: Hewlett-Packard, Texas Instruments, SUN Microsystems, Symbolics, Digital Equipment Corporation, and others.
- Installation of new releases of VMS, UNIX, and ULTRIX on our VAX computers.
- Installations of new releases of UNIX and other operating system software on ISI's other host computers and workstations.

15. DARPA HEADQUARTERS SUPPORT CENTER

Research Staff:
Dan Pederson
Ray Mason
Dan Trentham

15.1 PROBLEM BEING SOLVED

The ISTO computer center environment requires sophisticated systems capable of providing services for program managers and analysts commensurate with the complexity of their tasks. The systems must be supported in a cost-effective manner, while maintaining a less than 2 percent downtime.

15.2 GOALS AND APPROACH

To provide cost-effective service, ISI devised a set of remote environment surveillance and system diagnosis tools. These include remote monitoring (from ISI) of the ISTO computer room's temperature, humidity, power, fire-suppression system, VAX system crashes, and computer room entry via an automatic reporting system that uses standard telephone lines. ISI personnel are able to observe the status of the ISTO computer center through a series of video cameras whose images are transmitted to ISI via the ARPANET. Access to the Sun server systems for system software maintenance and enhancement is provided via remote dial-in. Hardware problem diagnosis is also achieved via remote dial-in, where diagnostics can be run from ISI. ISI hardware technical staff can, upon analysis of the diagnostics, determine the failing module and with the help of selected ISTO personnel, perform module swaps.

15.3 SCIENTIFIC PROGRESS

During this period two Sun 3/180 systems sharing 1.2 gigabyes of disk space were added to the ISTO computer room to act as servers for the installed base of Sun workstations and additonal workstations being purchased.

To allow ISI personnel remote access to the VAXes and Sun servers, ISI designed electronic switches to allow the consoles to be switched between a terminal and a modem. The modem telephone lines are secured through the LeeMah dial-up security system.

Implementation of dual porting on the RA81 disk drives on the VAX/11-780 was also accomplished during the reporting period. This will allow the VAX/11-750 access to the

VAX/11-780 disks, which means that processing can be switched to the VAX/11-750 when a crash of the VAX/11-780 is detected. The switch is nearly transparent to the user.

15.4 MILITARY IMPACT

ISI has proven that remote maintenance for small computer facilities is feasible and cost effective. ISI has been able to maintain a high mean time between failures for the existing systems through effective electronic and video surveillance of the computer room environment and remote access to the VAX and Sun systems in the computer room for maintenance and diagnostic purposes.

The "remote" approach to computer management of small unattended computer facilities could be a cost-effective solution for military applications.

15.5 FUTURE WORK

Remote surveillance of the computer facility for maintenance and security purposes will continue. ISI is investigating the possibility of upgrading the remote video capability to allow ISI to receive video images closer to real-time. The scan/compress/transmit time causes a delay, which could be partially alleviated by a new design of the compression algorithm, although much of the delay is caused by network load.

The full implementation of the LeeMah dial-back security system is ready to be activated as soon as administrative issues are resolved.

As wider and more complex usage of the Sun workstations takes place, additional disk storage will be required. The installation of a third Sun server, currently being tested at ISI, and additional disk drives will take place in FY88. Additional Sun workstations are being ordered to fill the ever-increasing needs of the ISTO staff.

The revamping of the ISTO Ethernet to a "thinnet" schema to provide ISTO with a more maintainable and expandable network is scheduled for FY88.

16. EXPORTABLE WORKSTATION SYSTEMS

Research Staff:
Dale Chase
Jim Koda

Support Staff:
Manon Levenberg
Ray Mason
Dan Trentham

16.1 PROBLEM BEING SOLVED

The use of dedicated, high-performance workstations and associated file servers offers substantial advantages over terminals connected to timesharing mainframes. The ever-increasing availability of integrated software environments has greatly facilitated the management and tracking of complex contracts and budgets for DARPA's program managers.

ISI provides DARPA/ISTO with the systems installation, configuration, and management expertise to successfully integrate new workstations and software into the ISTO computing environment.

16.2 GOALS AND APPROACH

The primary goal of this project was to install and support a workstation environment at the DARPA-ISTO offices in Arlington, Virginia. This environment is also intended to provide a testbed for software and hardware. The environment is configured to accommodate a variety of workstations with varying capabilities. This environment allows a minimum set of services on each workstation:

- document preparation
- mail handling
- file storage and manipulation
- administrative activities (spreadsheets, calendars, etc.)

These services are provided either directly on the workstations, or on the central server (a VAX-11/750). All of the workstations are connected to a local Ethernet and to the ARPA-Internet via a BBN-maintained gateway, and by equivalent code on the VAX that functions as a gateway in the event of failure.

To provide the DARPA-ISTO program managers with direct exposure to delivered software, we work closely with the developers at other DARPA-sponsored centers to handle installation, customization, and problem resolution within the ISTO environment.

16.3 SCIENTIFIC PROGRESS

Over the last few years, ISI has been the primary supporter of the computing environment at DARPA/ISTO. We have installed numerous Sun workstations, upgraded the VAX file server at ISTO from the 11/750 to the 11/780, maintained and upgraded the Sun Operating System as new versions have been released by Sun, and installed, upgraded, and maintained numerous high-resolution laser printers and printer software. In addition to the standard UNIX environment that Sun provides with their workstations, we have purchased and installed various third-party software systems for ISTO's internal use. These packages include spreadsheet software: QCalc and NQCalc, integrated document preparation systems: Scribe and Interleaf, text editors: Gnu Emacs, and database management systems: Ingres and Unify. ISI has served in an advisory capacity to ISTO for evaluation of new software packages and technologies.

A significant part of the ISTO support effort has been to customize software packages and operating systems used at ISTO to meet their operating requirements. Some examples are:

- We worked to modify and to fix problems with the MM mail package that came from SRI.
- We worked on CMM, a version of MM that was specifically designed to run under UNIX. This version from Columbia University is running at ISI and ISTO.
- Software was put in place to include the ISTO VAXes in the process of automatically updating the ISI mail forwarding database. This ensures that mail addressed to an individual at the wrong ISTO or ISI host will be forwarded to the correct host/mailbox.
- We modified the Sun and VAX kernel with the TCP/IP "upgrade" by Van Jacobson to improve performance and fix bugs.
- The VAX-11/750 was upgraded to Berkeley UNIX version 4.3. The remainder of the Sun workstations slated to be file served for user files from the VAX were converted. These systems were also upgraded to Sun OS version 3.3. In addition, new Sun servers and Sun 3 workstations were installed at ISTO.
- We installed the Interleaf document processing system at ISTO (both WPS and UPS). ISI spent much time customizing and solving Interleaf problems for ISTO.
- The BBN Diamond system was installed on one of the ISTO Suns.
- GNU Emacs was installed on the ISI and ISTO Suns, and appropriate manuals were distributed.
- X Windows was installed on several ISI and ISTO Suns.
- A Sun IPC board was installed in one of the Sun workstations at ISTO, allowing Suns to have a window running IBM/PC DOS applications.
- PC-NFS was installed for Sun IPC systems at ISTO.

- Certain features of public domain software used at ISTO were modified, and the laser printer driver software was customized to fit the computing environment.

16.4 IMPACT

The early acceptance of the workstations now in place at ISTO and the ease of adoption by previous users of timesharing systems clearly demonstrates the application of this new technology to office and administrative environments. This project has created considerable interest among our other user communities, military contractors in particular, as a means of providing increased computing capacity and enhanced functionality.

It is clear that dedicated personal workstations are superior to remote timeshared resources. The major advantages are the greatly increased responsiveness of the system, the elimination of delays caused by the Internet, and the enhanced functionality provided by the large-format display and "mouse" device offered on the workstation.

The ability to integrate additional workstation models into the environment, allowing them to function in an integrated manner, will greatly enhance the portability and transfer potential of research efforts being carried out by most of the DARPA-sponsored groups around the country. The sponsors will be able to run the developed software on identical hardware without waiting for it to be adapted to another system.

16.5 FUTURE WORK

ISI will continue to maintain, support, and enhance the total computing environment at ISTO with additional software targeted at management requirements, and new hardware and system upgrades as ISTO personnel and operational requirements change. ISI will install new workstations, assess the ability of new workstation software to meet ISTO requirements, investigate alternatives to the current VAX file server (whose function can be assumed by current Sun file server technology), and continue to upgrade and refine the computing environment's network software. Because the computing environments at ISI and ISTO are very similar, ISI will be able to test and experiment with all hardware and software upgrades to the ISTO computer center locally before installation at DARPA. This will ensure minimal downtime at ISTO for future upgrades. Many of the software updates will be installed directly over the ARPANET, allowing the update process to occur without disrupting any of the systems operating requirements at ISTO.

ISI will design, deliver, integrate, and customize software in the ISTO computing environment that will allow document interchange between document processing packages such as Interleaf, spreadsheet software, and database software packages. The design specifications for this document interchange software will be developed with ISTO and will be subject to a pre-delivery design review. Software will also be delivered to ISTO that will allow documents developed using the Interleaf document processing system to be interchanged between Sun workstations and Apple MacIntosh II and IBM-PC computers.

ISI will support the DARPA/ISTO staff with a hot-line and electronic mailbox for all inquiries, requests, and problem reports. Messages received from DARPA/ISTO will be answered within 24 hours. If the response will take longer than 24 hours to complete, an estimate of the requisite time will be included in the reply. Incoming requests and inquiries will be tracked by ISI's Action Accounting System. Every week, or requested period, a report summarizing the status of each open ISTO inquiry/request will be sent to DARPA/ISTO. This reporting mechanism will be coupled with the Action Accounting System to provide DARPA/ISTO program managers with quality tracking information on open ISI action items.

17. NEW COMPUTING ENVIRONMENT

***Research Staff*:**	***Support Staff*:**
Dale Chase	Glen Gauthier
Jim Koda	Manon Levenberg
	Ray Mason

17.1 PROBLEM BEING SOLVED

For the past decade, the computing needs of computer science researchers have been provided on large-scale, timesharing machines. These researchers no longer find these timesharing machines a viable approach for their research. The high-speed and dedicated workstation has become the machine of choice. ISI's two DEC mainframes running TOPS-20 are at full capacity and have become very costly to operate. Additionally, they are not capable of supporting many of the current and proposed research efforts at ISI.

As techniques and expectations (especially in the area of Artificial Intelligence) have advanced, general-purpose machines are no longer able to provide the style of interaction and capabilities, in terms of throughput and address space, that are required to support current research efforts. The NCE project continues to:

- provide a very significant improvement in both the amount and the quality of available computing resources to the ongoing ISI research efforts through the use of dedicated personal workstations and higher capacity centralized processors and servers.
- provide for diverse access methods to the common set of services.
- fully support the use of special-purpose workstations and processors as required by individual research projects.
- free up capacity on the existing mainframes by offloading some common functions to central servers.

17.2 GOALS AND APPROACH

The ISI research community is made up of a diverse set of projects and cultures. In order to best support a community with such varied needs, we decided to implement a heterogeneous environment consisting of a variety of personal workstations, dedicated small mainframes, and continued use of our existing timesharing systems. The New Computing Environment was designed to consist of local processing nodes (workstations and multiuser mainframes) connected via a local network to a set of central servers. A certain minimum set of functions must be available to all nodes in this environment: mail, file manipulation (access and transfer), and Internet connectivity (Telnet). The environment provided by these nodes and servers conceals the fine-grain detail from outside users; users external to the environment need not know what workstation a particular user is on, unless they want to. At minimum, these services should be available via any of several communication media: local area network, Internet, and dialup access. The NCE provides additional support as needed to allow low-end workstations and terminals on our mainframes to use these services as easily as the fully capable workstations in the Sun, DLion, and Apollo class. Primary access to these services is via the DoD TCP/IP protocol suite, with some extensions and enhancements to support particular modes of interaction that are not yet supported (e.g., random file access and remote procedure call).

The use of these services by our existing mainframes has resulted in immediate improvements for all users, as many CPU-intensive activities have been offloaded to central servers. Making these services available from our mainframes also provides for their use from home terminals. Access for these devices is via dialup lines to our existing mainframes, or to the servers themselves.

The following services are (or will be) provided to all nodes in the environment:

- centralized mail service
- centralized file service
- document formatting
- high-quality printing
- communications
- specialized processing

While the principal incentive for the New Computing Environment was the support and integration of a diverse set of dedicated personal workstations, not everyone at ISI requires this kind of hardware and capability. Until there is an actual need to migrate research efforts to workstations, there will be a significant population content with our existing mainframes (KLs and VAXes). However, the enhanced facilities offered by the provision of centralized servers also benefits this population. These enhancements take the form of increased accessibility of files and mail, increased reliability in terms of both data integrity and security, and improved performance of the existing mainframes due to the offloading of CPU-intensive activities to the dedicated servers.

To reduce the operating and maintenance costs of in-house computers, we have moved groups of users from TOPS-20 to VAXes running both UNIX and VMS. The TOPS-20 capacity freed by these transitions has been made available to other DARPA contractors, designated by the program managers at ISTO.

17.3 SCIENTIFIC PROGRESS

The main computing resource at ISI, a DEC VAX 8650, was updated to run Berkeley UNIX, version 4.3. We installed a number of Sun-3 workstations and file servers purchased under the second round of NCE acquisitions. Connectivity with the Xerox file server and printer was achieved using the XNS support in 4.3 bsd UNIX. A new version of the TOPS-20 operating system was installed to support access to the Imagen laser printers via the Internet.

The initial design and development of a system that makes existing archived files available over the Ethernet to any other system was completed and successfully installed. Work was completed on the integration of the Massbus Ethernet Interface System (MEIS) into TOPS-20. This TOPS-20-based resource is available on the local area network at ISI.

The MEIS allowed us to send packets over the local Ethernet. It took time and effort to obtain good interactive response and operation. The interface was then moved from the test machine to A.ISI.EDU, the operational system that would be its final home. The move caused some hardware problems, but they were quickly corrected.

The addition of the MEIS to A.ISI.EDU required an upgrade to version 6.1 for that system, which is a root domain server for the Internet. Work was done to integrate the domain server code into the 6.1 environment. A side benefit is that the A.ISI.EDU user community has been ushered into the domain name world, instead of being restricted to communicating with hosts in a static host table.

In other areas, work has been ongoing to improve the connectivity between ARPANET, MILNET, and ISINET systems. Different systems on the ISINET have been configured and tested to act as private gateways between the ISINET and the MILNET, and dramatic improvments have been observed. Routing strategies designed to make the most of such gateways are being tested.

Several improvements were made to the PC integration into the ISINET. Sun PC/NFS version 2.0 was received and installed on local IBM/PCs. A new release of the CMU PC/IP was obtained and installed on local PCs, employing a network-based installation scheme called Bootp to allow previously networked PCs to be updated without the need to shuffle floppies. Local improvments to the PC/IP code were folded in, including an enhanced lpr client that allows files to be spooled from PCs to a UNIX server for printing to any of the printers in the ISI environment.

Another package from CMU, the CMU/TEK TCP/IP package for VMS, was obtained and has been installed on local VMS systems, both VAX-11/750s and Micro-VAXes.

This brings full Internet connectivity to a segment of the ISI systems that had previously lacked TCP/IP access.

17.4 IMPACT

Our experience with personal workstations has made it clear that they represent a useful alternative to large, timeshared mainframes for certain research activities. Several projects within the institute have already moved the majority of their work to workstations, and have therefore been able to continue research that exceeded the capacity of our mainframes.

As the reliability of added facilities and of the entire environment is proven at ISI, we offer the same type of environment to outside contractors via the Exportable Workstation Systems project. These two projects combine to provide an ideal environment for developing and refining the facilities and capabilities that are becoming increasingly important in the command and control context. These capabilities include the handling of redundant databases and the support of a heterogeneous collection of hardware.

17.5 FUTURE WORK

We will identify the computing needs for the remainder of the user community that has yet to make the transition from centralized computer resources to workstations, and then plan configurations and procurements to meet those needs. We will also study subnet configurations of the local area network to optimize network traffic. To aid in this study, we plan to implement an extended network traffic analyzer that will provide detailed short-term logs and long-term traffic histograms.

The Laser Disk Archive project will be completed and installed at ISTO. We plan to study approaches that will allow remote workstations to gain access to network-based resources via dial-up lines and high-speed dedicated links.

We will also specify and implement a full-function mail service for IBM-PCs. The service will be integrated into the local area network and will interface to Internet mail services.

18. STRATEGIC COMPUTING DEVELOPMENT SYSTEMS

***Research Staff*:**
Dan Pederson
Stephen Dougherty

18.1 PROBLEM BEING SOLVED

Using recent advances in workstation/server architecture, artificial intelligence, and computer science, DARPA plans to create a new generation of "machine intelligence technology." The DARPA Strategic Computing program will be targeting key areas where advances can be leveraged towards broad advances in machine intelligence technology and the demonstration of applications of the technology to critical problems in defense.

The Strategic Computing program will be supported by a technology infrastructure. This infrastructure will serve to bootstrap the program by providing current state-of-the-art computing technology, network communications, and shared resources to the program participants.

The aim of this project is to provide development computers for the Strategic Computing program. System integration and the distribution of the systems to the program participants, as well as a defined architecture for system communications and resource sharing among the computers acquired, are requisite for this project.

One of the project's aims is to provide cost-effective and state-of-the-art engineering workstations to the Strategic Computing community. As the research in this program has progressed, and as the sophistication and complexity of available hardware and software in the vendor community have evolved, the more generalized workstations have proven to be a valuable adjunct to the already established base of specialized symbolic processors.

18.2 GOALS AND APPROACH

A number of machines have been developed to support high-speed processing of large symbolic programs. Each of these machines provides an extensive interactive programming environment, a sophisticated display-manager "window system," a real-time, window-oriented editor, incremental compilers, and dynamic linking.

These systems are the state of the art in program development environments. They are widely used in the research community for systems development, expert systems, natural language systems, mapping applications, and support of CAD/CAM environments.

Since many of these are single-user systems, they cannot be timeshared in the traditional sense and thus the cost per researcher is high. To bring these costs down, workers are currently placed in the awkward situation of having to schedule their computing needs. The resultant scheduling conflicts naturally lead to low researcher productivity.

An examination of dynamic machine use shows that many activities do not require the high-speed processing capabilities afforded by these machines. Less expensive, general-purpose machines supporting the same language base have become available only over the last few years. While formerly adequate only for less-intensive research activities such as editing, text preparation, and file scanning, these general-purpose machines are now sufficient for most of the more intensive program development and execution activities.

The duality of resources required for different types of activities and the availability of machines of appropriate cost/performance for carrying out those activities suggests a workstation/server architecture based on these two classes of machines and an interconnecting network. In order to support dynamic source code exchange between server and workstation, it is necessary for each to support the same AI language system. Common LISP is the natural choice in such an architecture, as it was designed with the concepts of commonality and portability as primary goals.

A significant fallout of such an architecture, when viewed as workstations and servers on TCP/IP-based networks, is the ability of researchers to communicate over the Internet in order to use high-powered resources available on prototype and low-production machines. This has been particularly useful on prototype machines being developed under the machine architecture phase of the program.

ISI has acquired a mix of these machines to support the requirements of the Strategic Computing program, and has helped to integrate and test individual systems as required for particular Strategic Computing program participants.

The integration tasks have avoided duplication of effort among Strategic Computing participants. ISI has collected software to support this architecture as it has been developed by vendors and the research community. ISI does a modest amount of testing, modifying, and augmentation of software; provides support for software exchange and distribution among DARPA development sites; helps assure working compatibility of the Common LISP systems; works with commercial vendors to resolve vendor-specific problems; and works to allow network compatibility of the systems with DoD network protocols (TCP/IP). The software and system integration efforts are carried out at the ISI Marina del Rey facility.

18.3 SCIENTIFIC PROGRESS

During the reporting period, additional symbolic-processing and general-purpose configurations were acquired. Some were installed and integrated at ISI, while most were shipped out to research facilities, educational sites, and government agencies.

In the past year, 75 machines were acquired as part of this Strategic Computing project, as well as a large variety of component peripherals, software packages (from Interleaf, CCA, Unipress, etc.), and software maintenance agreements. The major machines acquired were:

- 30 - Symbolics - 3645/3675
- 10 - Texas Instruments - Explorer
- 35 - Sun - various models
- Various hardware peripherals packages

Several of these machines are in use at ISI in preliminary work on natural language and the Common LISP Framework effort.

18.4 IMPACT

The DARPA Strategic Computing program will continue to develop and integrate advanced computer technology into military applications. Technological advancement will be sought in high-speed symbolic machines, as well as application of this technology to the military. Potential military application of this technology includes autonomous systems (land, air, and sea), battlefield management and assessment, planning and advising, and simulation systems.

The initial program applications included an autonomous land vehicle, a pilot's associate, and a carrier battle group battle management system. These applications test the evolving technology in the areas of vision, expert systems, speech recognition, and high-performance knowledge-processing systems. Each application seeks to demonstrate the new technologies' potential for providing a major increase in defense capability and to reflect a broad demonstration base that is relevant to each of the three services.

The development systems acquired through this project will support the technology base and the targeted applications under the Strategic Computing program.

18.5 FUTURE WORK

Several manufacturers have perceived that there will be a substantial marketplace for a more general-purpose symbolic processing engine in the future. We expect that the price/performance ratio of these new machines will continue to improve at a rapid pace, resulting in more efficient use of researcher time as more state-of-the-art machines can be acquired at the most reasonable price.

During the next year, ISI will continue to negotiate with vendors and acquire a mix of high-end LISP machines and other engineering workstations capable of handling symbolic processing in addition to other applications. As the workstation/server architecture is defined by DARPA, ISI will configure systems, test vendor software, and distribute the systems to the Strategic Computing program participants.

19. STRATEGIC C3 SYSTEM EXPERIMENT SUPPORT

***Research Staff*:**
Dan Pederson
Stephen Dougherty
Victor Ramos

19.1 PROBLEM BEING SOLVED

DARPA has defined an experiment in Strategic C3 systems to be conducted in cooperation with the World Wide Military Command Control System (WWMCCS) System Engineer (WSE) and the Strategic Air Command (SAC). The concept of the experiment is to demonstrate and evaluate the use of new technologies (such as the ARPANET, packet radio, network security, and distributed knowledge base techniques) for strategic command, control, and communication.

ISI's portion of the plan is to provide an initial core of necessary facilities (ARPANET/MILNET access, host systems, various software tools, Network Services support, etc.) to allow SAC personnel to gain experience with this technology and to ensure the success of the experiment. Specifically, SAC must have fairly broad-based experience with ARPANET-based online interactive computing. Installation and maintenance of modems and of 60 or more interactive CRT terminals, user training, system software training and support, and on-site maintenance of equipment for the 1000+ users at SAC Headquarters at Offutt Field, Omaha, Nebraska, and at a number of distributed SAC bases, are part of the continuing program.

19.2 GOALS AND APPROACH

The total specific goals of this DARPA experiment are to:

- demonstrate and evaluate the survivability of multinode computer-communication networks, including the ARPANET and packet radio, especially for remote access from both airborne platforms and surface sites.
- explore replication and reconstitution of critical knowledge bases on this network in the face of a loss of a large number of links.
- demonstrate and evaluate the rapid reconstitution of a network by rapid deployment of packet radio nets to reestablish connectivity between surviving elements of the network.
- conduct experiments and exercises to evaluate the utility of such a network on a distributed knowledge base to support post-attack C3 activities.

The experiment is defined as having three phases:

1. Phase I is planned to demonstrate air-to-surface packet radio links and gateways into the ARPANET/MILNET as a first step in evaluating the feasibility of a truly survivable strategic network.
2. Phase II is directed toward creating a survivable message system and databases through multiple copies of the critical components and data across the ARPANET/MILNET.
3. Phase III will address the feasibility of rapid reconstitution of a strategic network by deployment of packet radio networks to reconnect surviving elements of the network.

19.3 SCIENTIFIC PROGRESS

Since the inception of this project, ISI has been contracted to provide sufficient resources as the SAC user community increased. Additional Terminal Access Controllers (TACs) were installed to support the growing user community; additional lines and modems were also added to the datacommunications systems at various Air Force bases, in order to allow more widespread participation in the experiment via the MILNET. This equipment is also maintained by ISI. Upgrades that were installed to E.ISI.EDU, a Digital Equipment Corporation model KL-2060 host computer, increased SAC user productivity.

To support the VAX van effort, an experiment testing the mobility and reconstitutability of a partially destroyed network, new resources were obtained and configured into the systems.

19.4 IMPACT

One of the premises of this experiment is that SAC personnel will become more proficient and knowledgeable users of the current computer technology, available within the ARPANET/MILNET arena. This knowledge will allow SAC to make more effective evaluations of new technologies for strategic use, to identify areas of potential future research, and to implement those technologies found appropriate.

19.5 FUTURE WORK

ISI will continue to assist DARPA in planning this program, working to satisfy the communication and computer resource requirements of SAC Headquarters. In particular, ISI will do the following:

- continue to provide on-site maintenance support for the required equipment.
- continue to plan and assist in implementing improved SAC connectivity to the MILNET/DDN/ARPANET.

- maintain terminals and communication equipment for the connection of several Air Force bases to the MILNET, allowing increased participation in the experiment.
- continue to supply computer, programming, and administrative support to users at SAC Headquarters via the resources of the E.ISI.EDU computer, the Systems Programming, Operations, and Network Services staff in Marina del Rey, and the on-site technician.
- conduct a full equipment inventory.

ISI's most visible direct support will continue to be the on-site technician at SAC, who will be responsible for the identification of system malfunctions and for primary maintenance of on-site equipment. He will be supplied with required spare parts and will have maintenance contracts with the equipment vendors. Further support will be available from ISI in terms of additional spare parts, systems expertise, and documentation as necessary. ISI and the on-site maintenance technician will also be responsible for the off-site terminals at Vandenberg Air Force Base, Barksdale Air Force Base, March Air Force Base, and other locations as dictated by the requirements of the experiment. The on-site technician will coordinate data communications requests of SAC AD with SRI International and SAC under the supervision of ISI management. The technician will provide training to SAC personnel with backup from ISI as required.

ISI will provide program planning assistance to DARPA. We will continue to investigate the data communications requirements for SAC Headquarters, the HERT effort, and the ACCESS system (SAC Automated Command and Control Executive Support System).

RECENT PUBLICATIONS

1. Arens, Y., "Understanding language within the context model," *Communication and Cognition -- AI* 4, (2-3), 1987, 171-187. Special issue on artificial intelligence and discourse analysis.

2. Arens, Y., J. Granacki, and A. Parker, "Phrasal analysis of multi-noun sequences," in *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, ACL, Stanford, California, July 1987.

3. Arens, Y., J. Granacki, and A. Parker, "Understanding system specifications written in natural language," in *IJCAI 87: Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, IJCAI, Milan, Italy, August 1987.

4. Ayres, R., *FUSION: A New MOSIS Service*, USC/Information Sciences Institute, RR-87-194, October 1987.

5. Balzer, R., T. E. Cheatham, Jr., and C. Green, "Software technology in the 1990's: Using a new paradigm," in W. W. Agresti (ed.), *New Paradigms for Software Development*, pp. 16-22, IEEE Computer Society Press, 1986. Reprinted from *IEEE Computer*, November 1983, pp. 39-45.

6. Balzer, R., N. M. Goldman, and D. S. Wile, "Operational specification as the basis for rapid prototyping," in W. W. Agresti (ed.), *New Paradigms for Software Development*, pp. 116-132, IEEE Computer Society Press, 1986. Reprinted from *ACM SIGSOFT Software Engineering Notes*, December 1982, pp. 3-16.

7. Balzer, R., "Transformational implementation: An example," in W. W. Agresti (ed.), *New Paradigms for Software Development*, pp. 227-238, IEEE Computer Society Press, 1986. Reprinted from *IEEE Transactions on Software Engineering*, January 1981, pp. 3-14.

8. Balzer, R., "Living in the next generation operating system," in *Proceedings of the 10th World Computer Congress '86*, IFIP, Dublin, Ireland, September 1986.

9. Balzer, R., and N. M. Goldman, "Principles of good software specification and their implications for specification languages," in N. Gehani and A. McGettrick (eds.), *Software Specification Techniques*, pp. 25-39, Addison-Wesley, 1986.

10. Bisbey, R., II, R. Parker, and E. R. Cole, "An inexpensive megabit packet radio system," in *Proceedings of COMPCON Spring '86*, IEEE, San Francisco, March 1986. Also published as USC/Information Sciences Institute, RS-86-166, October 1986.

11. Booth, D., *Multiple Strongly Typed Evaluation Phases*, USC/Information Sciences Institute, RS-86-165, October 1986.

12. Cohen, Donald, "Automatic compilation of logical specifications into efficient programs," in *AAAI-86: Proceedings of the 5th National Conference on Artificial Intelligence*, pp. 21-25, American Association of Artificial Intelligence, Philadelphia, August 1986. Also published as USC/Information Sciences Institute, RS-86-175, November 1986.

13. Cohen, Danny, and J. Finnegan, "AI as the ultimate enhancer of protocol design," in *Proceedings of the Third Annual Artificial Intelligence and Advanced Computer Technology Conference*, Long Beach, California, April 1987. Also published as USC/Information Sciences Institute, RS-87-193, July 1987.

14. Cohen, Danny, "A mathematical approach to computational network design," in E. E. Swartzlander (ed.), *Systolic Signal Processing Systems*, chapter 1, Marcel Dekker, 1987.

15. Cumming, S., and R. Albano, *A Guide to Lexical Acquisition in the JANUS System*, USC/Information Sciences Institute, RR-85-162, February 1986.

16. Cumming, S., *Design of a Master Lexicon*, USC/Information Sciences Institute, RR-85-163, February 1986.

17. Cumming, S., *The Lexicon in Text Generation*, USC/Information Sciences Institute, RR-86-168, October 1986.

18. Feather, M. S., "The evolution of composite system specifications," in *Proceedings, Fourth International Workshop on Software Specification and Design*, pp. 52-57, IEEE Computer Society Press, Monterey, California, April 1986.

19. Feather, M. S., "An incremental approach to constructing, explaining, and maintaining specifications," in *Iteration in the Software Process: Proceedings of the 3rd International Software Process Workshop*, pp. 137-140, IEEE Computer Society Press, Breckenridge, Colorado, November 1986.

20. Feather, M. S., "Session summary: Next steps," in *Iteration in the Software Process: Proceedings of the 3rd International Software Process Workshop*, pp. 129-132, IEEE Computer Society Press, Breckenridge, Colorado, November 1986.

21. Feather, M. S., "Program specification applied to a text formatter," in N. Gehani and A. McGettrick (eds.), *Software Specification Techniques*, pp. 289-301, Addison-Wesley, 1986.

22. Feather, M. S., "A survey and classification of some program transformation approaches and techniques," in L. G. L. T. Meertens (ed.), *Program Specification and Transformation: Proceedings of the IFIP TC2/WG 2.1 Working Conference on Program Specification and Transformation*, pp. 165-195, North-Holland, 1987.

23. Feather, M. S., "Language support for the specification and development of composite systems," *ACM Transactions on Programming Languages and Systems*, April 1987.

24. Finn, G. G., *Routing and Addressing Problems in Large Metropolitan-scale Internetworks*, USC/Information Sciences Institute, RR-87-180, March 1987.

25. Harel, D., A. Pnueli, J. Pruzan-Schmidt, and R. Sherman, "On the formal semantics of statecharts," in *Proceedings of the Second Annual Conference on Logic in Computer Science*, Cornell University, June 1987.

26. Hovy, E., "Interpretation in generation," in *AAAI-87: Proceedings of the 6th National Conference on Artificial Intelligence*, American Association of Artificial Intelligence, Seattle, July 1987. Also published as USC/Information Sciences Institute, RS-87-186, April 1987.

27. Hovy, E.., "Generating natural language under pragmatic constraints," *Journal of Pragmatics* 11, 1987, 689-719.

28. Hovy, E., "What makes language formal?," in *Proceedings of the Ninth Conference of the Cognitive Science Society*, pp. 959-964, Seattle, 1987.

29. Johnson, W. L., "Specification via scenarios and fragments," in *Third International Software Process Workshop*, ACM SIGSOFT and IEEE-TCSE, November 1986.

30. Johnson, W. L., "Modeling programmers' intentions," in J. Self (ed.), *Intelligent Computer-Aided Instruction*, Chapman and Hall Press, 1986.

31. Johnson, W. L., *Intention-Based Diagnosis of Novice Programming Errors,* Morgan Kaufmann, Los Altos, California, 1986.

32. Johnson, W. L., and E. Soloway, "PROUST: An automatic debugger for Pascal programs," in G. Kearsley (ed.), *Artificial Intelligence and Instruction: Applications and Methods*, Addison Wesley, 1986.

33. Johnson, W. L., "Understanding and debugging novice programs," *Artificial Intelligence,* 1987.

34. Kaczmarek, T. S., R. Bates, and G. Robins, "Recent developments in NIKL," in *AAAI-86: Proceedings of the 5th National Conference on Artificial Intelligence*, American Association of Artificial Intelligence, Philadelphia, August 1986. Also published as USC/Information Sciences Institute, RS-86-167, November 1986.

35. Kasper, R. T., "Systemic grammar and Functional Unification Grammar," in *Proceedings of the 12th International Systemic Workshop*, Ann Arbor, Michigan, August 1985. Also published in USC/Information Sciences Institute, RS-87-179, May 1987.

36. Kasper, R. T., "A unification method for disjunctive feature descriptions," in *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, ACL, Stanford, California, July 1987. Also published as USC/Information Sciences Institute, RS-87-187, April 1987.

37. Kasper, R. T., *Conditional Descriptions in Functional Unification Grammar*, USC/Information Sciences Institute, Technical Report RR-87-191, November 1987.

38. London, P., and M. Feather, "Implementing specification freedoms," in C. Rich and R. Waters (eds.), *Readings in Artificial Intelligence and Software Engineering*, Morgan Kaufmann, 1986. Also published in *Science of Computer Programming 2*, 1982, 91-131, North-Holland.

39. Mac Gregor, R., and R. Bates, "The Loom knowledge representation language," in *Proceedings of the Knowledge-Based Systems Workshop*, St. Louis, Missouri, April 1987. Also published as USC/Information Sciences Institute, RS-86-188, May 1987.

40. Mann, W. C., *Dialogue Games*, USC/Information Sciences Institute, RR-79-77, October 1979.

41. Mann, W. C., and S. A. Thompson, "Relational propositions in discourse," *Discourse Processes* 9, (1), January-March 1986, 57-90.

42. Mann, W. C., and S. A. Thompson, "Antithesis: A study in clause combining and discourse structure," in R. Steele and T. Threadgold (eds.), *Language Topics: Essays in Honour of Michael Halliday*, John Benjamins Publishing Company, Amsterdam, 1987. Also published as USC/Information Sciences Institute, RS-87-171, April 1987.

43. Mann, W. C., and S. A. Thompson, "Rhetorical Structure Theory: Description and construction of text structures," in G. Kempen (ed.), *Natural Language Generation: Recent Advances in Artificial Intelligence, Psychology, and Linguistics*, Kluwer Academic Publishers, Boston/Dordrecht, 1987. Proceedings of the Third International Workshop on Text Generation, August 1986, Nijmegen, The Netherlands. Also published as USC/Information Sciences Institute, RS-86-174, October 1986.

44. Mann, W. C., "Text generation: The problem of text structure," in D. McDonald and L. Bolc (eds.), *Natural Language Generation Systems*, Springer-Verlag, New York, 1987. (In the series *Symbolic Computation*.) Also published as USC/Information Sciences Institute, RS-87-181, March 1987.

45. Mann, W. C., and S. A. Thompson, *Rhetorical Structure Theory: A Theory of Text Organization*, USC/Information Sciences Institute, RS-87-190, June 1987.

46. Mann, W. C., and S. A. Thompson, "Rhetorical Structure Theory: A framework for the analysis of texts," in A. Duranti and B. Schieffelin (eds.), *IPRA Papers in Pragmatics,* Volume 1, 1987. Also published as USC/Information Sciences Institute, RS-87-185, April 1987.

47. Matthiessen, C., "Representational issues in Systemic Functional Grammar," in *Proceedings of the 12th International Systemic Workshop*, Ann Arbor, Michigan, August 1985. Also published in USC/Information Sciences Institute, RS-87-179, May 1987.

48. Matthiessen, C., and S. A. Thompson, "The structure of discourse and 'subordination'," in J. Halman and S. A. Thompson (eds.), *Clause Combining in Discourse and "Subordination"*, John Benjamins Publishing Company, Amsterdam, 1987. Also published as USC/Information Sciences Institute, RS-87-183, April 1987.

49. Matthiessen, C., "Semantics for a systemic grammar: The chooser and inquiry framework," in M. Cummings, J. Benson, and W. Greaves (eds.), *Systemic Perspectives on Discourse*, John Benjamins Publishing Ccmpany, Amsterdam, 1987. Also published as USC/Information Sciences Institute, RS-87-189, May 1987.

50. Matthiessen, C., "Notes on the organization of the environment of a text generation grammar," in G. Kempen (ed.), *Natural Language Generation: Recent Advances in Artificial Intelligence, Psychology, and Linguistics*, Kluwer Academic Publishers, Boston/Dordrecht, 1987. Proceedings of the Third International Workshop on Text Generation, August 1986, Nijmegen, The Netherlands. Also published as USC/Information Sciences Institute, RS-87-177, April 1987.

51. Mostow, J., and W. Swartout, "Towards explicit integration of knowledge in expert systems: An analysis of MYCIN's therapy selection algorithm," in *Proceedings, AAAI-86: Fifth National Conference on Artificial Intelligence*, pp. 928-935, American Association of Artificial Intelligence, Philadelphia, August 1986. Also published as USC/Information Sciences Institute, RS-86-176, November 1986.

52. Najjar, W., and J.-L. Gaudiot, "Reliability and performance modelling of hypercube-based multiprocessors," in *Proceedings of the Second International Workshop on Applied Mathematics and Performance/Reliability Models of Computer/Communication Systems*, Rome, Italy, May 1987. Also published as USC/Information Sciences Institute, RS-87-184, April 1987.

53. Najjar, W., and J.-L. Gaudiot, "Distributed fault-tolerance in data-driven architectures," in *Proceedings of the Second International Conference on Supercomputing*, Santa Clara, California, May 1987.

54. Najjar, W., and J.-L. Gaudiot, "Multi-level execution in data-flow architectures," in S. K. Sahni (ed.), *Proceedings of the International Conference on Parallel Processing*, St. Charles, Illinois, August 1987.

55. Najjar, W., J.-L. Jezouin, and J.-L. Gaudiot, "Parallel execution of discrete-event simulation," in *Proceedings of the 1987 International Conference on Computer Design*, IEEE, Port Chester, New York, October 1987.

56. Najjar, W., J.-L. Jezouin, and J.-L. Gaudiot, "Parallel discrete-event simulation on multiprocessors," *IEEE Design and Test* 4, (6), December 1987, 41-44.

57. Najjar, W., and J.-L. Gaudiot, "A hierarchical data-driven model for multigrid problem solvers," in *Proceedings of the International Symposium on High-Performance Computer Systems*, Paris, France, December 1987.

58. Pavlin, I., "Motion from a sequence of frames," in *Proceedings, SPIE Conference on Mobile Robots '87*, pp. 204-213, Cambridge, Massachusetts, November 1987.

59. Postel, J., G. G. Finn, A. R. Katz, and J. Reynolds, *The ISI Experimental Multimedia Mail System*, USC/Information Sciences Institute, RR-86-173, September 1986.

60. Robins, G., "The ISI Grapher: A portable tool for displaying graphs pictorially," in *Proceedings of Symboliikka '87*, Helsinki, Finland, August 1987. Also published as USC/Information Sciences Institute, RS-87-196, September 1987.

61. Smoliar, S. W., "A view of goal-oriented programming," in *Proceedings of COMPCON Spring '86*, pp. 112-117, IEEE Computer Society Press, 1986.

62. Smoliar, S. W., "Operational requirements accommodation in distributed system design," in W. W. Agresti (ed.), *New Paradigms for Software Development*, pp. 133-139, IEEE Computer Society Press, 1986. Also published in *IEEE Transactions on Software Engineering,* November 1981, pp. 531-537.

63. Smoliar, S. W., "Specifications and transformations: When does the work *really* get done?," in L. G. L. T. Meertens (ed.), *Program Specification and Transformation: Proceedings of the IFIP TC2/WG 2.1 Working Conference on Program Specification and Transformation*, pp. 483-489, North-Holland, 1987.

64. Smoliar, S. W., "Review of *Induction: Processes of Inference, Learning, and Discovery,*" *IEEE Expert* 2, (3), 1987, 92-93.

65. Smoliar, S. W., "Review of *Conceptual Structures: Information Processing in Mind and Machine,*" *Artificial Intelligence,* 1987. Also published as USC/Information Sciences Institute, RS-87-182, March 1987.

66. Sondheimer, N. K., and B. Nebel, "A logical-form and knowledge-base design for natural language generation," in *AAAI-86: Proceedings of the 5th National Conference on Artificial Intelligence*, American Association of Artificial Intelligence, Philadelphia, August 1986. Also published as USC/Information Sciences Institute, RS-86-169, November 1986.

67. N. K. Sondheimer, ed., *Proceedings of the Strategic Computing Natural Language Workshop*, USC/Information Sciences Institute, SR-86-172, May 1986. Proceedings of a workshop held at Marina del Rey, California, May 1-2, 1986.

68. Swartout, W., "Knowledge needed for expert system explanation," *Future Computing Systems*, 1986. Revised version of a paper published in *AFIPS Conference Proceedings,* National Computer Conference, pp. 93-98, 1985.

69. Swartout, W., and R. Neches, "The shifting terminological space: An impediment to evolvability," in *Proceedings, AAAI-86: Fifth National Conference on Artificial Intelligence*, American Association of Artificial Intelligence, Philadelphia, August 1986.

70. Swartout, W., "Explanation by design," in *Proceedings of the Second Annual Conference on Artificial Intelligence and Advanced Computer Technology*, 1986.

71. Swartout, W., "Beyond XPLAIN: Toward more explainable expert systems," in *Proceedings of the Congress of the American Association of Medical Systems and Informatics*, 1986.

72. Swartout, W., and R. Balzer, "On the inevitable intertwining of specification and implementation," in W. W. Agresti (ed.), *New Paradigms for Software Development*, IEEE Computer Society Press, 1986. Reprinted from *Communications of the ACM*, July 1982. Also published in N. Gehani and A. McGettrick (eds.), *Software Specification Techniques,* pp. 41-45, Addison-Wesley, 1986.

73. Swartout, W., and S. W. Smoliar, "On making expert systems more like experts," *Expert Systems* 4, (3), August 1987.

74. Swartout, W., "Explanation," in Shapiro (ed.), *Encyclopedia of Artificial Intelligence*, John Wiley and Sons, 1987.

75. Swartout, W., and S. W. Smoliar, "Explaining the link between causal reasoning and expert behavior," in *Proceedings of the Symposium on Computer Applications in Medical Care*, Washington, D. C., November 1987. To appear in P. L. Miller (ed.), *Topics in Medical Artificial Intelligence,* Springer-Verlag.

76. McKeown, K., and W. Swartout, "Language generation and explanation," in *Annual Review of Computer Science,* Annual Reviews, Inc., 1987.

77. Tanner, W., *SUN Manual for ISI Users*, USC/Information Sciences Institute, RR-87-198, September 1987.

78. Tung, Y., C. Matthiessen, and N. K. Sondheimer, *On Parallelism and the Penman Natural Language Generation System*, USC/Information Sciences Institute, RR-87-195, September 1987.

79. Wile, D. S., "Organizing programming knowledge into syntax-directed experts," in *International Workshop on Advanced Programming Environments*, Trondheim, Norway, June 1986.

80. Wile, D. S., "Program developments: Formal explanations of implementations," in W. W. Agresti (ed.), *New Paradigms for Software Development*, pp. 239-248, IEEE Computer Society Press, 1986. Also published in *Communications of the ACM,* November 1983, pp. 902-911.

81. Wile, D. S., and D. G. Allard, "Worlds: An organizing structure for object-bases," *ACM SIGPLAN Notices* 22, (1), January 1987. Proceedings of the ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments.

82. Wile, D. S., "Local formalisms: Widening the spectrum of wide-spectrum languages," in L. G. L. T. Meertens (ed.), *Program Specification and Transformation: Proceedings of the IFIP TC2/WG 2.1 Working Conference on Program Specification and Transformation*, pp. 459-481, North-Holland, 1987.

83. Yen, J., "Can evidence be combined in the Demster-Shafer theory?," in *Proceedings of the AAAI Workshop on Uncertainty in Artificial Intelligence*, pp. 70-76, Seattle, Washington, July 1987.

84. Yen, J., "Implementing evidential reasoning in expert systems," in *Proceedings of the AAAI Workshop on Uncertainty in Artificial Intelligence*, pp. 180-188, Seattle, Washington, July 1987.